1

WRDC-TR-89-2095

HIGH TEMPERATURE TRIBOMETER

Forest J. Carignan
Bruce H. Knoth
Advanced Mechanical Tech., Inc.
151California Street
Newton, MA 02158

June 1989

Final Report for Period Sep 88 to Mar 89

Approved for public release; distribution unlimited.

AERO PROPULSION AND POWER LABORATORY
WRIGHT RESEARCH DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6563

NOTICE

This technical report has been reviewed and is approved for publication.

[illegible]

CHRISTOPHER J. KLENKE, PROJECT ENG
Lubrication Branch
Fuels and Lubrication Division
Aero Propulsion and Power Laboratory

[illegible]

PHILLIP W. CENTERS, Acting Chief
Lubrication Branch
Fuels and Lubrication Division
Aero Propulsion and Power Laboratory

FOR THE COMMANDER

[illegible]

BENITO P. BOTTERI, Chief
Fuels and Lubrication Division
Aero Propulsion and Power Laboratory

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) 2236-1	5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR-89-2095
6a. NAME OF PERFORMING ORGANIZATION Advanced Mechanical Tech., Inc. 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Aero Propulsion and Power Lab (WRDC/POO) Wright Research Development Center
6c. ADDRESS (City, State, and ZIP Code) 151 California Street Newton, MA 02158	7b. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB OH 45433-6563
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Department of the Air Force 8b. OFFICE SYMBOL (If applicable) PMRSA	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-88-C-2871
8c. ADDRESS (City, State, and ZIP Code) Air Force Systems Command Aeronautical Systems Division Wright-Patterson AFB, OH 45433-6503	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 65502F PROJECT NO. 3005 TASK NO. 21 WORK UNIT ACCESSION NO. 03

11. TITLE (Include Security Classification)
HIGH TEMPERATURE TRIBOMETER

12. PERSONAL AUTHOR(S)
Carignan, Forest J. and Knoth, Bruce H.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 9/8/88 TO 3/7/89	1989, June	80

16. SUPPLEMENTARY NOTATION
This is a Small Business Innovative Research Program, Phase I report.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Tribological research has become more important due to the awareness that friction and wear accounts for a large portion of our nation's energy and raw material usage. A machine is available that performs pin-on-disk friction and wear tests at temperatures up to 1200°C. Software to control the machine and acquire test data using IBM-PC compatible computers was designed and developed and initial tests of the software were conducted. A commercial software package, Labtech Notebook, was used as the basis for the system, although unique data-acquisition and graphics modules were also developed. A piston type liquid lubricant feed pump was chosen for accurately dispensing small quantities of test lubricant. A prototype of a solid powder feed system was built and functioned well with granular material but unsatisfactorily with the two very fine powders. (AU)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassifed	
22a. NAME OF RESPONSIBLE INDIVIDUAL Christopher J. Klenke	22b. TELEPHONE (Include Area Code) 513-255-9654	22c. OFFICE SYMBOL WRDC/POO

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

PREFACE

This research was sponsored by the United States Air Force under the Defense Small Business Innovation Research (SBIR) Program, Air Force Contract Number F33615-88-C-2871. This report covers the Phase I work conducted during the period of September 1988 to March 1989

TABLE OF CONTENTS

1.0 INTRODUCTION 1

1.1 Background 1

1.2 Description of Existing Tribometer 2

1.3 Problem Definition 3

1.3.1 Computer Control. 4

1.3.2 Wear Rate Measurements 4

1.3.3 Lubricant Injection 5

1.4 Phase I SBIR Study 6

2.0 SOFTWARE 7

2.1 Software Goals 7

2.2 Computer and Interface Hardware Requirements 8

2.3 Software Operation 8

2.3.1 Software Installation 11

2.3.2 Starting the Software 12

2.3.3 Manual Control of the Tribometer. 12

2.3.4 Conducting a Test. 17

2.4 Software Development 17

2.4.1 The Code Development Environment 18

2.4.2 Machine Control 19

2.4.3 Data-Acquisition 22

2.4.4 Graphics. 27

2.4.5 Commercial Data-acquisition Code 36

3.0 WEAR MONITORING. 40

4.0 LUBRICANT DELIVERY SYSTEM. 46

4.1 Liquid Lubricant Delivery 46

4.2 Solid Lubricant Delivery. 47

4.3 Furnace Modifications 51

5.0 CONCLUSIONS. 53

5.1 Software. 53

5.2 Wear Monitoring 53

5.3 Lubricant Injection 54

6.0 RECOMMENDATIONS 55

6.1 Software. 55

6.2 Wear Monitoring 55

6.3 Lubricant Feed 55

APPENDIX I Description of Previously Built High Temperature Tribometer 56

APPENDIX II Source Code. 63

APPENDIX III Liquid Lubricant Pump 72

LIST OF TABLES

Table 2.2.1 DT2801A Channels 9

Table 2.2.2 DDA06 Channels 10

Table 2.3.1 Key Combination for Tribometer Control 14

LIST OF FIGURES

Figure 2.3.1 Main Screen of FW.EXE Showing Three Windows. 13

Figure 2.3.2 Setpoint and Gain Windows in FW.EXE 16

Figure 2.4.1 Data-Flow Diagram for Data-Acquisition Module 23

Figure 2.4.2 Complete Linked List with Two Graph Objects 32

Figure 3.1 Block Diagram of Tribometer Slides 43

Figure 4.1 Feed System 48

1.0 INTRODUCTION

1.1 Background

The field of tribology has received considerable attention in the last few decades due to the increased awareness that friction and wear accounts for a large portion of our nation's energy and raw material usage. What was generally once thought to be an area governed more by experience than by science, has become an important area of research. Investments of time and money produce significant improvements in the tribological properties that affect the systems and materials in use today.

Time is an important issue to tribologists because the lifetime of a typical system usually extends over many hours or years. Testing for improved wear resistance in the actual system can therefore take even longer. Methods for accelerating testing are extremely valuable to the researcher and a variety of tribology test machines are available. The effectiveness with which they simulate actual conditions varies; often known materials are tested to provide a baseline for other materials. If the wear mechanism is not significantly altered from the real system to the test machine then the results are generally useful.

More tribology testing has probably been performed relative to internal combustion engines than to any other device. The actual wear mechanisms involved are complex and not likely to be exactly duplicated in anything but an actual engine. This is expensive and time consuming. However, there are specific areas of an engine in which certain forms of wear predominate and can be independently tested. At Top Dead-Center (TDC) the piston velocity goes to zero and the hydrodynamic film separating the ring from the cylinder wall disappears. Here we have boundary lubrication where surfaces contact and testing of materials in a sliding test apparatus with and without lubricants can give useful tribological information.

Duplication of actual operating temperatures during testing is desirable, since material and lubricant properties are highly temperature dependent. These temperatures are being raised through the use of ceramic engine components and new lubricants aimed at increasing engine efficiency and life.

Methods and machines for testing materials under simulated operating conditions are therefore necessary and the temperature requirements are out of the operating regime of ordinary tribology equipment.

1.2 Description of Existing Tribometer

AMTI has, over a period of years, developed a tribometer which is currently available as a standard product and forms the basis for the proposed machine. A description is provided in Appendix I. The most important features of the standard machine include 1200^{O}C operation and controlled atmosphere test conditions. A discussion of features generally useful in a tribometer are provided here.

There is no universally accepted test sample geometry for tribology testing, but pin-on-disk or plate type equipment is the most common. This is due to the fact that it can be simple and readily implemented. Many machines are inverted drill presses with dead-weight loaded pins mounted in a friction force measuring device.

Optimally, material testing results should not be test machine sensitive; but due to equipment variables such as vibrations, this is not generally the case. The simplest machines are also not readily suited for elevated temperatures, controlled atmospheres and remote control and monitoring.

Good control over the normal specimen loading should be provided by the machine. Dead weights are often used but they are not readily controllable or variable, and they have mass. This last factor produces variability in loading due to specimen deflections accelerating the mass in the normal load direction. It is generally good practice to monitor the normal load during testing, even though it is apparently fixed.

The friction force due to sliding should be continuously measured. This is optimally done in conjunction with the normal force measurement via a two-axis force transducer. The two independent electrical signals can be recorded and sent to an analog division circuit to provide a continuous indication of the friction coefficient. This compensates for variations in normal force and can also be done by computer with the transducer signals.

The specimen holders and transducer should be as stiff as possible to reduce vibrations. It is sometimes advantageous to angle the pin specimen with respect to the plate by a degree or so such that an angle less than 90° is formed in the direction of pin movement. Stick/slip and frictional vibration can sometimes be eliminated in this manner. However, if oscillatory motion is required, this technique is not suitable and perpendicular specimens should be used. In all cases, vertical runout of the pin specimen should be minimized and some adjustment is usually required and should be provided.

The specimens should be easily removable and firmly clamped in the test machine without damage. A range of specimen sizes should also be accommodated.

In addition to unidirectional sliding, oscillatory motion of the plate specimen in an arc greatly increases the versatility of the machine by providing tests with zero sliding-velocity conditions at the end points. This is especially important for lubricant tests. Good control over rotational and oscillatory motion should be provided.

The test atmosphere can be well controlled through the use of a totally sealed chamber. Evacuation of the test chamber and backfilling with the test gas of choice is the best method for providing a known atmosphere. If capable of operation in a vacuum, low outgassing materials should be used throughout the machine.

Elevated temperature testing is generally provided by electrical resistance heating elements. Below $1200^{\circ}C$ in air Kanthal Al winding can be used. Induction heating has no inherent temperature limitations other than the life of the susceptor and heated components in the atmosphere of operation. In a vacuum, arcing due to high voltage must be prevented.

There are several requirements in addition to longevity of machine components at high temperatures. The heat-up and cool- down times should be as short as possible. The specimen holders should firmly clamp their parts in spite of thermal expansions and permit easy removal. Other machine features such as stiffness and force measurement should not be compromised more than necessary.

1.3 Problem Definition

There are three important improvements that are needed on the standard machine in order to perform the required tests. They are:

- Computer control of the machine along with a data acquisition system. An IBM-PC is used.

- Wear rate measurement via displacement transducer on pin holder.

- Lubricant injection system for small quantities of liquid and solid lubricants.

1.3.1 Computer Control

Electrical signals are provided for all outputs on the standard machine. Interfacing these to a computer requires an A/D converter with suitable pre-signal conditioning electronics to put the signals into a typical $\pm$ 10 volt range. Many signals need no further processing; thermocouple, pressure sensor, and force transducer signals must be amplified. There are no significant barriers to be overcome as far as computer inputs are concerned.

Similarly, electrical signals control almost all functions of the machine in order to enable remote operation with the bell jar closed. Many functions are manually switch controlled and would need to be connected to a computer controlled relay panel. The manual control option should not be lost.

A software package is required that offers control of the tribometer on an immediate basis, allowing the operator to turn signals on and off and define setpoints easily. Furthermore, the package must acquire data, display it as it is collected, and store it for further analysis.

In this phase of the project the software has been designed and written and some preliminary testing has been completed. The software appears to work correctly and reliably, however further testing of the software is required. This testing is not possible until a tribometer is available under the second phase of this contract.

1.3.2 Wear Rate Measurements

Wear rate measurements on test specimens are generally performed after completion of the test and removal from the machine. Measurements of very small amounts of wear are possible by profilometry and optical measurement techniques on plates and pins respectively. During pin-on-disk-testing, it is possible to measure wear by monitoring the wear depth into the pin and plate. A displacement transducer which measures axial pin displacement during testing will measure moderate to large amounts of wear. Resolutions on the order of a few millionths of an inch can be achieved with commercially available LVDT's, but compensation must be made for thermal expansion effects. These are significant; at 1000^{o}C the plate and pin holder surfaces have moved together more than .05" (.13 cm) from their room temperature positions. Optimistically assuming the temperature profiles can be controlled to vary less than 1^{o}C from their steady-state values, direct measurement of pin displacement will resolve no less than 10 microns of axial wear. This is a moderate but not small amount of wear. The Phase I work in this area is mainly an analysis of the errors which will limit the accuracy of the wear measurement.

A differential measurement device which eliminates most of the thermal expansion errors could be envisioned. The measurement device would have to function in the heated zone and would therefore most likely be capacitive or optical. However, no practical method for doing so has been found.

1.3.3 Lubricant Injection

The standard machine has been provided in the past with a lubricant injection system. A peristaltic pump with a variable flow rate was used to inject liquid lubricants onto the test specimens. The pump was located outside the test chamber such that 5 CC of liquid was required to fill the feed line. No provision was made for automatic feeding of solid lubricants.

In this program very small quantities of lubricants require the dispensing device to be located near the test specimens. A positive-displacement type device is necessary in order to accurately meter the lubricant. This applies to both liquids and powders but a powder dispensing system is of a different design than a liquid system.

Single separate systems which will feed any liquid or solid are likely to be quite difficult to find or develop.

Some furnace changes are required for use with lubricants. The disposal of small amounts of used lubricants can be accomplished by storing them on a disc holder plate which has vertical sides to form a type of cup.

1.4 Phase I SBIR Study

The effort described in this report was sponsored by the Department of the Air Force under contract No. F33615-88-C-2871 as a Phase I study in the overall Defense Small Business Innovation Research (SBIR) Program. The principal goal of this investigation was to improve and automate a high-temperature pin-on-disk tribometer. The specific major objectives were as follows:

1. Design, develop and test software required to control the tribometer and gather data during tests.

2. Investigate and devise a practical means for wear rate measurement during testing.

3. Design and test a solid and liquid lubricant injection system.

The Phase I effort was a software and hardware development program to evaluate the feasibility of specific approaches to the above goals. The major portion of the program was devoted to designing, building, testing, and evaluating software and hardware needed for improving the existing tribometer. The work done on the project is presented and discussed in this report.

2.0 SOFTWARE

The standard tribometer includes electrical connectors and switches that allow most of its functions to be controlled by computer; furthermore, data may be collected by the computer through the same connectors.

The advantages of automated data-collection and control are well known. The data is collected and stored automatically, freeing the operator to perform other tasks. The chances of human error are minimized, and the bias that may result from reading gauges, charts, or timepieces is eliminated. Furthermore, the data may be converted to meaningful units and displayed immediately, either in alphanumeric or graphical form, allowing the operator to evaluate the test results and their quality as the test proceeds.

The computer can also monitor the test system and sound an alarm or shut the system down if certain signals exceed specified levels. The computer performs some actions automatically, allowing the operator to perform a complicated task simply by pressing a button on the keyboard.

2.1 Software Goals

The tribometer system is used in research laboratories on many types of projects. The purpose of the software is to simplify the use of the tribometer and automate data-collection, thereby reducing the effort required to complete a friction-and-wear test and improving the quality of the results.

The challenge of creating a software package for research environments is to provide enough flexibility for the users to reach their goals without being limited by the software. The package must be easy to modify and have a wide range of capabilities so it can be used for many different projects.

A second, almost contradictory, criteria for the software is that it be easy to use. The package must have the flexibility to meet the needs of the advanced researcher, but it must be usable by technicians or others who may conduct tests but have minimal knowledge of computer systems.

These requirements almost dictate the use of either a configurable software package with a large amount of flexibility or a programmable package. Either type of system allows a skilled individual to define and set-up test processes which a less-knowledgable person may then operate. Such systems also allow access to the data during the test for monitoring and after the test for analysis and comparison with other sets of data.

2.2 Computer and Interface Hardware Requirements

The software is designed to run on IBM-PC compatible computers. The requirements are that the computer have 640K of RAM and a hard disk. A math coprocessor is not required but is recommended to increase the speed of the calculations. The computer must have either an EGA or VGA color graphics card and monitor. The VGA, with its higher resolution, is recommended. Additional monitors may be supported in the future.

Two interface cards are required for the system. One is a Data Translation DT2801A data-acquisition card. This is a card with sixteen channels multiplexed to a single analog-to-digital converter. These channels are used to acquire data from the system. It also has two digital-to-analog channels which are used to set the positions of the upper and lower slides on the system. Furthermore, it has sixteen digital I/O channels, but they are not currently used in the friction and wear system. Table 2.2.1 lists the A/D and D/A channels on the DT2801A.

The second interface card is a Metrabyte DDA-06. This card has six channels of digital-to-analog conversion and 24 digital I/O channels. Table 2.2.2 lists the function of each of the channels on the DDA-06.

2.3 Software Operation

The software package is built around a commercial product, Labtech Notebook (available from Laboratory Technologies Corporation, Wilmington, Ma). Labtech Notebook is a sophisticated data-acquisition package, and meets all the requirements for the data-acquisition section of the software. Labtech Notebook, however, does not have the capability of controlling the tribometer in an immediate mode; it does not allow the operator to easily turn switches on and off or set output levels.

A second program, called FW and written by AMTI, provides the immediate control

Table 2.2.1

DT2801A Channels

A/D Channel	Function
0	Friction Force
1	Normal Force
2	Friction Coefficient
3	Torque
4	Upper Slide Position
5	Amplified Upper Slide Position
6	Lower Slide Position
7	Amplified Lower Slide Position
8	Displacement
9	Tachometer
10	Rotary Shaft Position
11	Humidity
12	Temperature
13	Chamber Pressure
14	Chamber Vacuum
15	Spare

D/A Channel	
0	Lower Slide Position Setpoint
1	Upper Slide Position Setpoint

Table 2.2.2

DDA06 CHANNELS

D/A Channel	Setpoint For	Signal Range
0	Temperature	0-10 V => 20 to 1200 C
1	Load	+/-10 V => +/-10 lb
2	Oscillator offset	+/-10 V => +/-180 Degrees
	or Rotary Speed	+/-10 V => +/-1000 RPM
3	Oscillator Frequency	0-10 V => 0.1 to 5 Hz
4	Oscillator Amplitude	0-10 V => 0 to +/-180 Deg
5	Spare	

Digital I/O Channel	Controls	Action
PA0	Zero Friction Force Amplifier	Lo - Operate, Hi - Zero
PA1	Zero Normal Force Amplifier	Lo - Operate, Hi - Zero
PA2	Zero Torque Amplifier	Lo - Operate, Hi - Zero
PA3	Zero Displacement Transducer	Lo - Operate, Hi - Zero
PA4	Friction Force Gain	Lo - 1 lb/V, Hi - 1 N/V
PA5	Normal Force Gain	Lo - 1 lb/V, Hi - 1 N/V
PA6	Torque Gain	Lo - 100 in-oz/V, Hi - 1 NM/V
PA7	Displacement Gain	Lo - .01 in/V, Hi - .001 in/V
PB0	Raise Upper Slide	Lo - off, Hi - on
PB1	Lower Upper Slide	Lo - off, Hi - on
PB2	Upper Slide Automatic	Lo - off, Hi - on
PB3	Raise Lower Slide	Lo - off, Hi - on
PB4	Lower Lower Slide	Lo - off, Hi - on
PB5	Lower Slide Automatic	Lo - off, Hi - on
PB6	Heater Enable	Lo - off, Hi - on
PB7	Load Enable	Lo - off, Hi - on
PC0	Spindle Enable	Lo - off, Hi - on
PC1	Rotary/Oscillatory Mode	Lo - rotary, Hi - Oscillatory
PC2	Sine/Triangle Wave	Lo - Sine, Hi - triangle
PC3	Spare	
PC4	Spare	
PC5	Spare	
PC6	Spare	
PC7	Spare	

that the operator needs. This program allows the major functions of the tribometer to be controlled by pressing buttons on the computer keyboard. The heater may be toggled on or off, for instance, by pressing ^H (control-H). FW communicates with Labtech Notebook to provide a continuous display of the input signals so that the state of the machine can be monitored while changing its configuration. This program provides immediate, straight-forward control of the system, but it does not have true data-acquisition and storage capabilities. It relies on Labtech Notebook for data-acquisition and storage.

Labtech Notebook has the required data-acquisition features. It allows the user to acquire data, store it, and graph it as it is collected and users can easily modify it to meet their data-acquisition requirements. Furthermore, it can be configured to change collection rates based upon a sudden change in the input data. For instance, it can be configured to increase the sampling frequency if the derivative of the friction force suddenly increases. Once a user has defined the configuration, it can be saved and used for all subsequent runs. The test process can be automated so a less-skilled operator can run tests that were previously defined and Labtech Notebook easily interfaces to commercial data-analysis packages such as Lotus 1-2-3 or BBN's RS/1.

Notebook's shortcoming is that it does not easily allow immediate control of the tribometer. However, it does have provisions to allow another program to operate in tandem with it, so a program was created that interacts with Notebook and provides the required immediate control of the system.

This program, called FW, has been integrated with Notebook so that pushing the "P" key from Notebook's main menu automatically starts it up. Once this program is running, the operator has instantaneous control over many features of the tribometer.

2.3.1 Software Installation

The software is delivered in three parts. The first two are Labtech Notebook and a companion real-time-access routine, and the third part is the FW package from AMTI. Labtech Notebook must be installed first. It should be installed according to the instructions in its manual and tested by going through its tutorial.

The second part is the "real-time access" package with Labtech. This should also be installed following the instructions in its manual. This package is required to allow the FW package to communicate with the DT2801A board.

The FW package is installed by copying the contents of the FW disk to the NB directory on the hard disk. The files provided on the FW disk are program and batch files that allow the FW program to interact with Labtech Notebook. A setup file for Notebook is also provided that defines the setup needed to acquire data from the tribometer.

2.3.2 Starting the Software

The software is started by typing "NB" at the DOS prompt. This starts the Notebook package and displays the main notebook menu. Now all the functions of Notebook are available as well as the FW program for manual control of the machine.

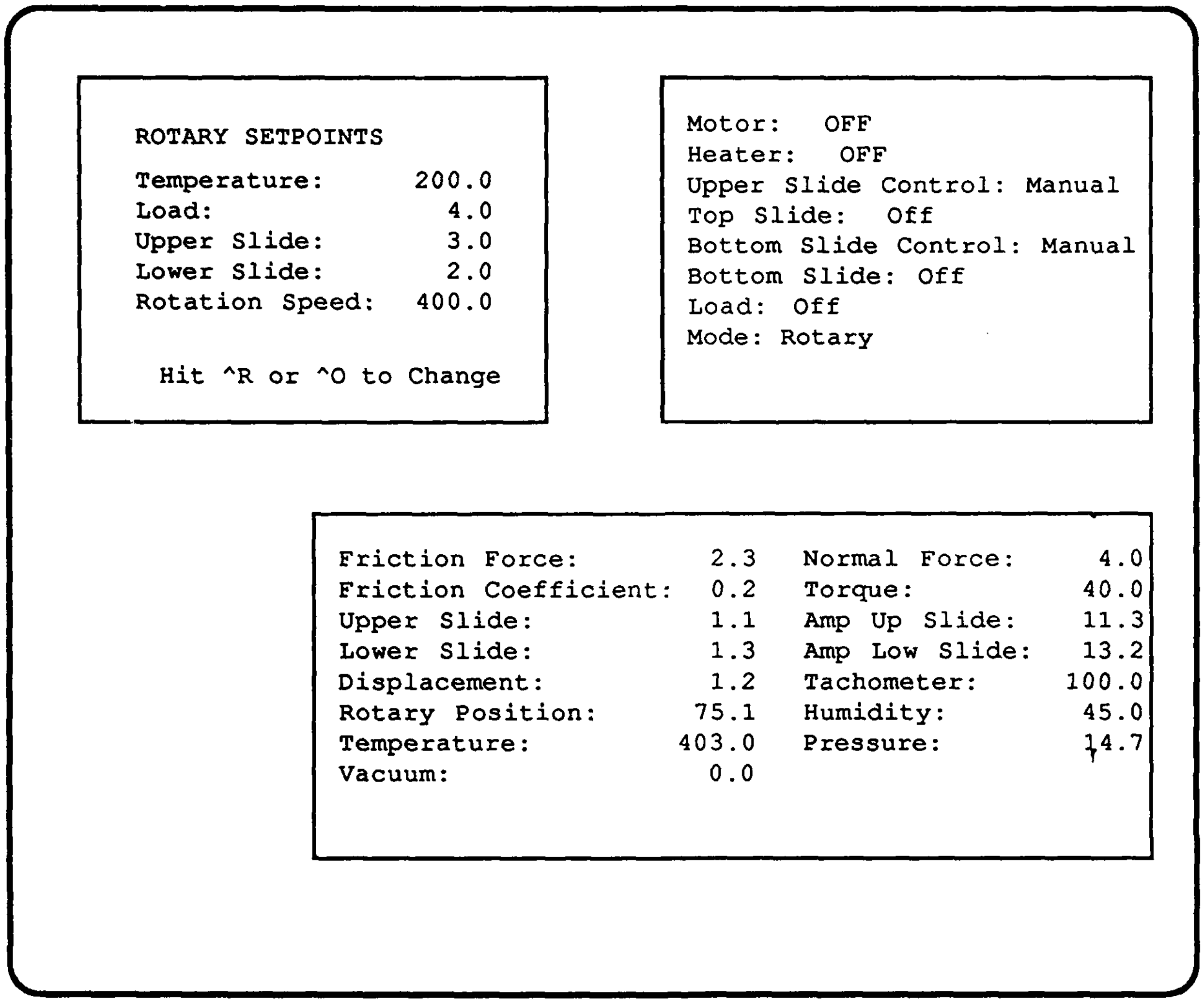
Before running a test, the tribometer must be set-up. This is done through the FW program.

2.3.3 Manual Control of the Tribometer

To run the FW program from Notebook, press "P" (for program) when Notebook's main menu is visible. After a few seconds, the display screen for FW appears. This screen has three visible regions (Figure 2.3.1). One region shows the status of various tribometer switches. Another region shows the setpoints for the system, and the third region displays the signal levels from the tribometer. Several additional windows appear during operation. One is a help window, another is for changing gains, and two more allow changing of setpoints.

Switching Components On or Off

All of the switch settings displayed can be controlled using the keyboard unless the change-gains or change-setpoints windows are visible. Specific key-combinations activate or deactivate system components. For instance, typing ^M (the notation ^M means control-M and is invoked by holding down the Ctrl key and pressing the M key) toggles the motor between off and on. When the motor goes on, it operates in the mode defined by the setpoints (described later).



```
ROTARY SETPOINTS

Temperature:       200.0
Load:                4.0
Upper Slide:         3.0
Lower Slide:         2.0
Rotation Speed:    400.0

  Hit ^R or ^O to Change
```

```
Motor:  OFF
Heater:  OFF
Upper Slide Control: Manual
Top Slide:  Off
Bottom Slide Control: Manual
Bottom Slide: Off
Load:  Off
Mode: Rotary
```

```
Friction Force:           2.3   Normal Force:       4.0
Friction Coefficient:     0.2   Torque:            40.0
Upper Slide:              1.1   Amp Up Slide:      11.3
Lower Slide:              1.3   Amp Low Slide:     13.2
Displacement:             1.2   Tachometer:       100.0
Rotary Position:         75.1   Humidity:          45.0
Temperature:            403.0   Pressure:          14.7
Vacuum:                   0.0
```

MAIN SCREEN OF FW.EXE SHOWING THREE WINDOWS

FIGURE 2.3.1

Pressing ^H turns the heater on or off. When on, the heater is enabled and the temperature rises to the value defined by the temperature setpoint.

^L turns the load on or off. When the load is on, it is set to the value defined by the load setpoint.

The top and bottom slides can be raised and lowered from the keyboard. The top slide is raised by pressing ^Q, and lowered by pressing Alt-Q (press the ALT key and then the Q key). As long as the keys are held down, the slide goes up or down. The bottom slide is raised using ^Z and lowered with Alt-Z. These key combinations are chosen because they graphically represent the action. On the old-style PC keyboards, the Ctrl key is above the Alt key, so it raises the slides and Alt lower the slides. Q, representing the top slide, is above the Z, which represents the lower slide. Unfortunately, the Alt and Ctrl keys have been moved on the new-style keyboards so the keys don't make graphic sense.

Table 2.3.1 lists the key combinations and their effects.

Table 2.3.1 - Key Combinations for Tribometer Control

^M	-	Toggles the motor
^H	-	Toggles the heater
^T	-	Toggles the top slide auto control
^B	-	Toggles the bottom slide auto control
^L	-	Toggles the load
^R	-	Defines a rotary test
^O	-	Defines an oscillatory test
A-Q	-	Lowers the top slide
^Q	-	Raises the top slide
A-Z	-	Lowers the bottom slide
^Z	-	Raises the bottom slide
^G	-	allows gain control

Note: ^Z means Ctrl-Z and A-Z means Alt-Z.

HELP SCREEN

A help screen, which lists all the information in Table 2.3.1 is available. Press the F1 key to turn it on, and press that key again to turn it off.

SETPOINT DEFINITION

The actions of the motor and the levels of the load and temperature are defined with setpoints. Setpoints are defined for either rotary or oscillatory tests. When rotary setpoints are defined the system is put in rotary mode automatically, and when oscillatory levels are defined the system is placed in oscillatory mode.

Press ^R to define rotary setpoints or press ^O to set oscillatory setpoints. Now a screen appears that lists the available setpoints and their levels (Figure 2.3.2). Using the arrow and number keys, move the highlight bar to the appropriate values and change them. When everything is correct press Q (for Quit) and the screen disappears. Now the setpoints are redefined, and if sections of the tribometer are enabled, the tribometer levels go to their new values. For instance, if the heater is on and the temperature setpoint is changed, the temperature changes after pressing "Q".

If the tribometer switches are off, the levels change after the switch is enabled. For instance, changing the load setpoint does not affect the load if the switch is off, but when the load is turned on by pressing ^L the load assumes the value defined by the setpoint.

The current values of all tribometer signals are displayed continuously. As the setpoints and switches are changed, these levels change correspondingly.

GAIN CONTROL

Each of the four signal conditioning amplifiers has two possible gain settings. The four amplifiers process the displacement transducer signal, the torque signal, the normal force signal, and the friction force signal. The gains for these amplifiers may be changed by typing ^G. Now a window appears that lists the four signals and the current gains (Figure 2.3.2). Any of the gains may be changed to its alternate setting by typing the first letter of its name. For instance, the Displacement Transducer gain may be

```
Furnace Temperature:   200.0 Deg C
Load:                   4.00 Lb
Rotary Speed:           200.0 RPM
Top Slide Position:     2.50
Bottom Slide Pos:      1.25
```

Rotary Setpoints Entry Window

(reached by pressing ^R)

```
Furnace Temperature:    200.0 Deg C
Load:                    4.00 Lb
Offset:                  75 Degrees
Frequency:               100.0 Hz
Amplitude:               45.0 Degrees
Top Slide Position:      2.50
Bottom Slide Pos:       1.25
```

Oscillatory Setpoints Entry Window

(reached by pressing ^O)

```
Displacement Gain:        100 V/in
Torque Gain:              0.0 V/in-oz
Friction Force Gain:      1.0 V/lb
Normal Force Gain:        1.0 V/lb

 Press First Letter to Change
```

Gain Window

(reached by pressing ^G)

SETPOINT AND GAIN WINDOWS IN FW.EXE

FIGURE 2.3.2

changed by typing "D". Once the gains are set, type Q, Esc, or ^G to return to the main screen. After the gains are set, the program modifies the RTSETUP.PRN file for Labtech Notebook to include the new gains.

ZEROING THE AMPLIFIERS

The four signal-conditioning amplifiers have an auto-tare feature. This feature is activated by pressing ^D for the Displacement Transducer amplifier or ^F for the force and torque amplifiers. When ^D or ^F is pressed, the signal to zero the appropriate amplifiers is switched to a high voltage and held for about two seconds, then it returns to its low value. This signal initiates zeroing of the amplifiers.

When you are done with the manual control portion of the program, press the F10 key to return to Notebook's main menu.

2.3.4 Conducting a Test

Tests are conducted in two phases. The first phase is to define the test parameters, and the second phase is to run the test and acquire data.

The first step is to define the data-acquisition channels and parameters. An initial setup file is provided with the software package; this package defines all the channels, sample rates, and the data-storage file. These parameters almost certainly have to be changed, however, for specific test conditions and durations. Change the parameters using Labtech Notebook.

Once the data-acquisition parameters have been defined, then the machine parameters have to be set. Assuming that the test pieces have been installed in the tribometer and the machine is ready to go, use the manual-control features of the software to define the load, temperature, etc. Then enable the heater, motor, and load and press the F10 key to exit the manual control section. Now press "G" (for Go) and the data-acquisition section starts.

2.4 Software Development

This section describes the software development process and the evolution of the package. One goal of this project was to develop a complete package specifically tailored to the tribometer, including data-acquisition, analysis, and graphics capabilities. Eventually it became apparent that the scope of the project would exceed the resources available, so an alternative approach was sought. The decision was made to adapt a commercial package to the requirements of the tribometer. Although the package would be somewhat more complex than custom software, the development time would be greatly reduced and the user would receive a more flexible system.

Labtech Notebook was chosen because it is a complete menu driven system that has many provisions for modifications and additions. The menus and data-acquisition parameters can be changed from outside the program, the data-acquisition portion can run in the background while another program operates in the foreground, and the data-acquisition board can be easily accessed from outside programs through a device driver.

The next sections of this report describe the development of the data-acquisition, machine-control, and graphics code. Although some of these packages did not end up in the final system due to the change in direction, they were undertaken as part of this project so they will be described here.

2.4.1 The Code Development Environment

The software was written on an 80386-based IBM-PC compatible. All of the code described was developed using the same general packages. Virtually all code is written in C using version 5.0 of the Microsoft C compiler (Microsoft Corp, Bellevue, WA). A small amount of the code was written in assembler.

The windows were created using a commercially available windows package (C Power Windows from Entelekon, Houston, Texas). Supplementary code developed previously at AMTI for a different project was also used.

The graphics were developed using the Halo package of graphics functions (from Media Cybernetics, Silver Spring, MD). This package was selected because it is device-independent and has been used successfully on previous projects. The graphics code did not end up in the final system due to the switchover to a commercial data-acquisition package.

Much of the source code is listed in the Appendices for completeness, but code developed on previous projects or obtained from outside vendors is not listed.

2.4.2 Machine Control

This section of the code led to the FW program that is included in the tribometer software package. It interacts with the Metrabyte DDA06 digital-to-analog conversion board to set output voltages and control the digital signals. This package is designed to allow the user to set the machine for their test requirements while removing some of the complexity of the tribometer. This package offers the user immediate control of the system in a more intuitive fashion than is available from the tribometer control panel. For instance, to raise a slide using the control panel the user must first check that the slide control is in "manual" mode, then the slide is moved with toggle switches. If the slide is in automatic mode, then the switches don't function. When the slide is raised via the computer, the status of the slide is not important. As soon as the user presses the key to raise the slide, the slide will move. If the slide had been in automatic mode, the computer will first switch it to manual mode without the user's knowledge. Subtle features like this make the system easier to use.

FW.C SOURCE CODE FILE

The main body of the machine control code is in the FW.C file. This file holds code which first reads in the configuration files and then jumps to the machine control function. The machine control function initializes the window displays, installs a keyboard interrupt function, and then enters a loop that updates the windows and watches for certain key strokes. The help display (activated by pressing the F1 key), and the rotary and oscillatory change screens are initiated from this loop. The loop is exited when the user presses the F10 key. Then the keyboard interrupt is restored and the windows are removed.

INPUT FILES

Two data files are used to define the display of data. The file FWINP.DAT lists the active input channels for the DT2801A and provides labels for the channels. When the FW program displays data on screen, it only shows the channels listed in the FWINP.DAT file and it displays each label along with the reading from that channel.

FWINP.DAT may list all sixteen channels, if they are all active, or it may list only two if just two channels are important. For instance, a two-channel FWINP.DAT may be

```
2 "Temperature"
4 "Load"
```

In this case, the FW program would only display values from two channels. It would read channel #2 (these are Labtech Notebook Channel numbers, not DT2801A channels) and display the label "Temperature" followed by the value from that channel. Then it reads channel #4 and displays "Load" followed by it's value. This file allows the user to customize the channel value display for his requirements.

The other file, FWOUTP.DAT, is used to define the D/A channels on the DDA-06. It holds labels and conversion factors for the eight possible setpoints (temperature, load, oscillator offset, rotary speed, oscillator frequency, oscillator amplitude, lower slide position, and upper slide position). The file must list values for these quantities in the order shown. A typical FWOUTP.DAT file is shown below:

```
"Temperature" 0.0 10.0 20.0 1200.0 409.6 "C"
"Load"      -10.0 10.0 -10.0 10.0 204.8 "lb"
"Oscillator Offset" -10.0 10.0 -180.0 +180.0 204.8 "Degrees"
"Rotary Speed" -10.0 10.0 -1000.0 1000.0 204.8 "RPM"
"Oscillator Frequency" 0.0 10.0 0.1 5.0 409.6 "Hz"
"Oscillator Amplitude" 0.0 10.0 0.0 180.0 409.6 "Deg"
"Lower Slide Position" 0.0 10.0 0.0 8.7 409.6 "in"
"Upper Slide Position" 0.0 10.0 0.0 8.7 409.6 "in"
```

The entries in one row are in the following order: value label, lowest output voltage, highest output voltage, lowest output value, highest output value, bits/volt for the D/A channel, and a label for the units on this channel. For instance, the first row in the file shown above means that it is a temperature channel. The D/A range spans 0 to 10.0 volts and the corresponding output values are 20.0 to 1200.0 degrees. The D/A has 409.6 bits per volt for this channel, and the units on the output values are C (ie. degrees C). The values provided are used to calculate the correct voltage and bit level for a given output value (ie. 200 degrees C corresponds to 1.53 volts and 625 bits).

KEYBOARD INTERRUPT ROUTINE

The keyboard interrupt routine is the heart of the machine control code. The address of this routine is inserted in the keyboard interrupt vector in the IBM PC (interrupt number 9), and the conventional keyboard interrupt routine is chained to the end of this routine.

Every keystroke is intercepted by the new interrupt routine. The scan code of the keystoke is examined, and if it is relevant to the real-time machine control code (ie. if it is the Ctrl, Alt, or a relevant letter key) it is acted on. If not, it is passed on to the original interrupt code so the PC can use it.

If the Ctrl or Alt key is struck, then a status flag for those keys is set to 1. If either of those keys is released, then the status flag is set to 0 and a command is sent to stop the slides from moving (regardless of whether they were moving). If an additional key is struck while the Ctrl or Alt key is held down, it is evaluated. If the key is not needed by the control routine, then it is passed on to the PC, otherwise appropriate action is taken. If a key other than Alt or Ctrl is pressed while those keys are not active, then the scan code is passed directly to the PC. In this manner, the keystrokes are received, evaluated, and used by the control code in real time.

The digital output signals on the DDA06 are turned on or off by sending a command to a port on the computer. Each of the eight-bit digital I/O ports on the DDA06 maps to an output port on the computer. Turning one of the digital signals on for a port requires sending an eight-bit value to the port with the corresponding bit high. The status of the eight signals can be found by reading the port.

The procedure for turning one output signal on for a given port is to first read the port to get its current status, then do a bitwise OR with the value to turn the appropriate bit on without affecting the other bits, and finally send the altered value back to the port to make the signal high.

The above procedure can be messy if written directly into the computer code. Therefore, a function called SET was created that makes the code more readable. With the SET function, a given component, such as the heater, can be turned on or off or toggled to the alternate state with function calls such as

set(HEATER,ON)	turns it on, and
set(HEATER,TOGGLE)	changes the heater's state.

The function is listed in the appendix in the file DDA06.C, and the codes that allow it to work with symbols such as HEATER and LOAD are listed in the header file FW.H.

2.4.3 Data-Acquisition

A realtime data-acquisition package was designed and developed to gather data from the tribometer. Although the first versions of this package worked after debugging, Labtech Notebook was used in place of this routine so this code is not used in the final version of this software.

The data-acquisition package is designed to gather data in realtime, convert the data to floating point values and invoke an arbitrary list of functions to process the data. Only the actual data-gathering is interrupt driven, the other sections of the code are run continually at a lower priority.

The realtime kernel is a short section of code written in assembler that attaches to the timer interrupt (address 1CH). The kernel, activated every time the interrupt fires, instructs the DT2801A to take one set of data and transfer it via DMA to a specified buffer. The kernel is an independent body of code that is driven by the timer interrupt. But it interacts with and is governed by the data-acquisition control code.

Figure 2.4.1 shows a simplified schematic of the data flow in the data-acquisition code. The program functions are denoted as rectangles, and data storage locations are shown as rectangles with rounded corners. The names in the rectangles, such as initialize_daq(), are the names of the functions (the pair of parentheses denotes a function call in C.) The function calling structure is shown with thick arrows, and the thinner arrows show the data flow between functions. This diagram is simplified to explain the structure of the code, the actual code (listed in the appendix) is more complex.

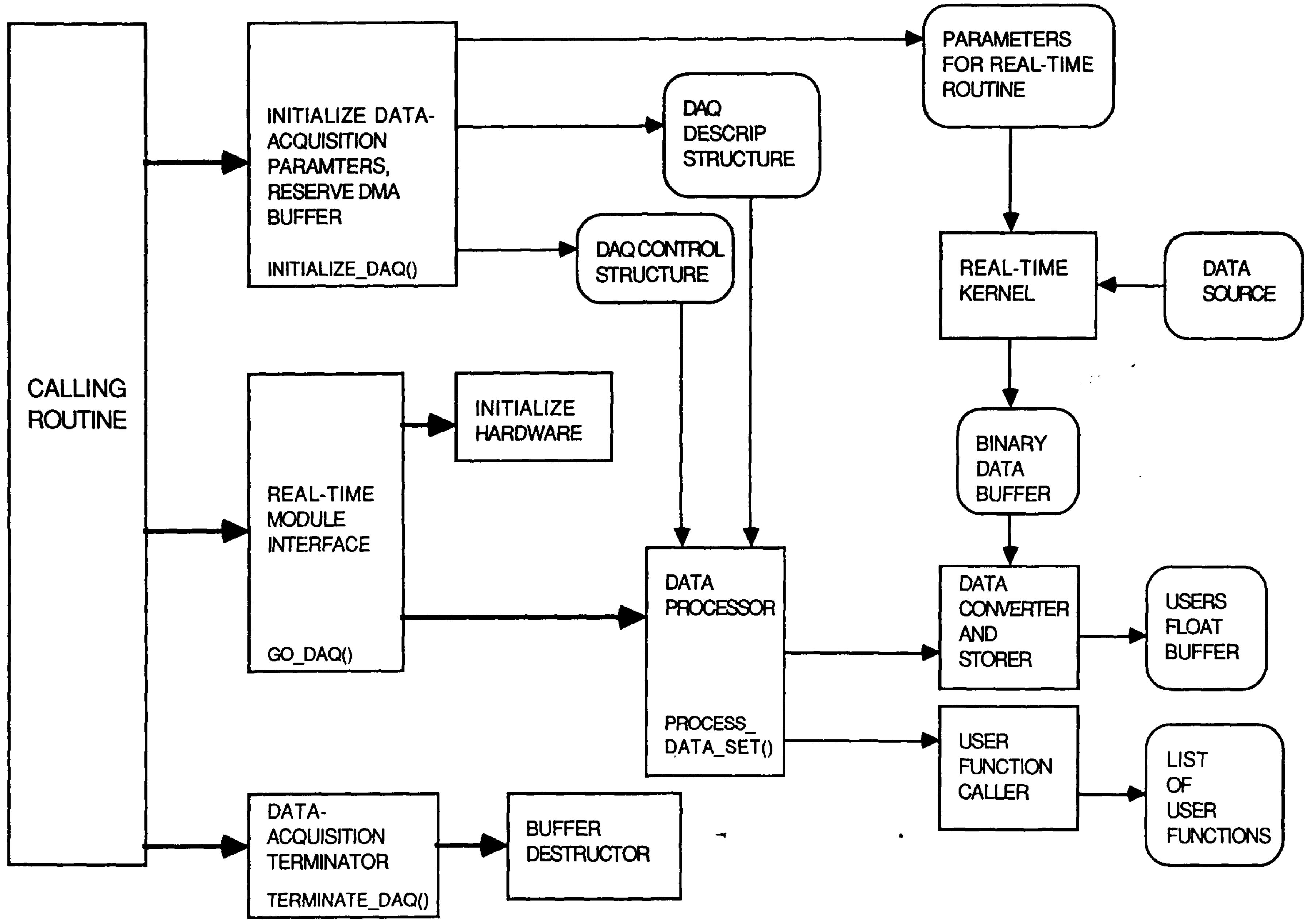


FIGURE 2.4.1 - DATA-FLOW DIAGRAM FOR DATA-ACQUISITION MODULE

The main calling routine, shown to the left of the diagram, is the main program. This program initializes, starts, and terminates data-acquisition through three separate function calls. First, it calls the initialize_daq() function and passes it information such as the start and end channels, the collection frequency, and the number of data samples required. This function, in turn, reserves a buffer for the collected binary data and fills two data structures. One structure, called daq_descrip, holds parameters that describe the collected data and should be saved with the data. The other structure, called daq_control, holds parameters needed to process the data but which don't have to be saved after that. The function also fills a buffer with information needed by the realtime kernel. Once the data-acquisition section has been successfully initialized, control passes back to the calling program.

The calling program is responsible for starting the actual acquisition. It may start collection immediately, or it may wait for a keystroke from the operator. It starts the acquisition process by calling the go_daq() function. The go_daq() function retains control until all data has been acquired and processed.

The go_daq() function operates in two phases. In the first phase, it initializes the DT2801A board, the DMA chip in the PC, and changes the timer tic to the rate required for data collection. Then it places the address of the data-acquisition kernel in the timer interrupt vector. Once the kernel has been installed, the go_daq() function enters its second phase of operation.

In the second phase, it monitors the parameters used by the kernel to see how many data sets have been collected and if data-acquisition has terminated. Once a new set has been received, the go_daq() function calls the process_data_set() function. This function converts the new set of data from binary to floating point form (in the correct units) and places it in the floating point buffer provided by the calling program.

After a data set has been converted, the next step is to call all functions in a list provided by the calling program. Each of these functions may have a unique role. For instance, one or more functions may drive a plotting routine which displays the data in graphical form. Another function may monitor the data to be sure it doesn't exceed preset conditions. These functions are optional, and they provide a means to use the data as it is collected without altering the data-processing code directly.

The time to process a data set may be greater than the data-acquisition period. In a non-interrupt driven process, this would be a problem since it would delay acquisition of subsequent data and throw off the data-acquisition schedule. The kernel, however, is interrupt-driven so it proceeds regardless of what the computer is doing. Although the previous set of data may be in the midst of processing, the kernel still collects data at the correct time. This independence is crucial to acquire data at precise time intervals, however it requires excellent communication between the kernel and the processing routine. For instance, the processing routine needs to know how many sets of data have been collected and placed in the binary data buffer. It can process data up-to and including the last set that has been fully collected, but not beyond that.

The kernel needs to know how many sets of data to collect, and it must shut itself down after they have been collected. If the kernel continues to operate after all the data has been collected, it will inevitably write data outside the designated buffer and the program will crash. Once the kernel has shut itself off, the processing program needs to recognize that data collection has stopped. Then it finishes processing the last sets of data and returns to the calling program.

There are many subtle problems that occur during development of code that is interrupt-driven. The first problems occur when installing the timer vector. If an interrupt occurs while the vector is being installed, the computer dies. Also, if there is a bug in the code that the vector points to, the computer dies as soon as the new vector is installed. If there are any coordination problems between the kernel and the processing program, the computer will almost certainly die or get caught in an infinite loop.

Both the kernel and the processing routine rely on comparing calculated addresses. Those calculations can easily be in erro. because the C compiler makes assumptions about addresses that the assembler does not. Therefore, tricks are required to force the compiler to calculate the same addresses that the assembler code does.

Another addressing problem typically arises in interrupt driven code. The 8088 microprocessor and its successors use a segmented addressing scheme. Rather than addressing RAM with one CPU register and allowing all memory to be addressed from 0 to N-1 (where N is the number of bytes of RAM in the computer), the 8088 uses two registers. One register is called a segment register. It is a sixteen bit register that holds a base address. The second register holds the offset within the current segment. The advantage of this approach is that the 8088 can create 20 bit addresses using two 16 bit registers. A 20 bit address is formed by shifting the segment address four bits to the left and then adding the offset, allowing the 8088 to address one megabyte of RAM.

There are several disadvantages to this method, however. One is that retaining and moving two sixteen bit values for each address consumes program storage and computer time. The significant problem for interrupt-driven processes, though, is that an interrupt can occur at any time so the segment registers may not be correct for the interrupt routine. Therefore, it is critical that the interrupt code includes provisions to change the segment registers to the proper values before processing any data. At the end of the interrupt routine, the registers must be returned to their original values. Failure to handle the segment registers properly causes the program to crash.

DMA (direct memory access) driven data-storage introduces some coordination problems. DMA is controlled by a chip that is independent of the microprocessor. DMA avoids the loss in performance that occurs when data is transferred from the data-acquisition card to memory via the CPU. Unfortunately, in the IBM PC, there is no indication that a DMA transfer is complete. This has several implications. First, a new data-collection cycle cannot be initiated while the DMA chip is busy so the top frequency of the data-collection routine must be limited to a value that lets the DMA chip complete its task. Secondly, the kernel initiates a DMA transfer and returns control to the main program. The DMA transfer, however, proceeds for an unknown length of time and, until it is complete, the current data set is not valid. Therefore, the kernel may indicate that a data set has been collected because it has intitiated DMA but the DMA transfer may not be complete. Furthermore, there is no indicator that shows when it is complete so a means to monitor the status of last set of data is needed.

A technique was developed to make sure a data-set is complete before processing it. Before data collection begins, each word in the data storage buffer is filled with the value 1010H. This number is a unique value that cannot be created by the data-acquisition card. (the card is twelve bits, so it is limited to values below 0FFFH). Before the data-processing code uses the last set of data, it checks the last value in that set. If it equals 1010H, then the DMA transfer is not complete. Otherwise, the transfer has finished.

Some problems can be avoided during development by writing the kernel as a program function that is called at a known time by the main program rather than by an interrupt. This technique allows the kernel's code to be tested and debugged and it also allows the processing code to be tested. Once the code seems to operate properly the kernel is converted to interrupt form and attached to the interrupt. Now problems with installing the kernel first appear and coordination problems between the kernel and the processing code arise. These problems do not occur when the kernel is called directly by the main routine because the calling sequence is fixed so coordination problems are minimized. When the kernel is called independently at an arbitrary rate many new coordination problems surface, both because data is acquired and processed at different rates and because the segment problems mentioned earlier first arise.

2.4.4 Graphics

A graphics system was developed that is device-independent and allows multiple graphs to be displayed on the screen. Device-independence allows one set of graphics code to work with any display or hard-copy device. For instance, the same code works with CGA, EGA, VGA and other monitors dot-matrix printers, and x-y plotters. Unfortunately, "device-independent" is a misnomer. A better term might be "quasi-device-independent" since a small section of device-specific code is required for each separate display, but virtually all the code is written with no knowledge of the details of the display.

Display of multiple graphs on the screen requires that display of a graph be independent of creating the graph. There may be one, two, three, or four plots visible at any one time, each in its section of the screen. If one plot is visible, then it fills the whole screen; if two plots are visible, then each fills half of the screen, etc. To have clean, workable, graphics code specification of the plot must be independent of display of the plot.

The graphics code uses a "normalized screen" and a "screen manager". The viewing area is normalized to the coordinate (0,0) for the lower-left corner and (100,100) for the upper-right corner regardless of the size of the screen or the size of the viewing area on the screen. The locations of the X and Y axes on the plot are fixed, and everything else is drawn relative to the axes. The x-axis spans the range (20,20) to

(90,20) and the range of the y-axis is (20,20) to (20,90).

One of the difficulties of creating graphs on a computer screen is labeling the plots. For this graphics code, labels for the plot are defined and rules are given for their placement. For instance, the X-axis label might be defined as "Time" and it is located as follows:

1. A reference point is defined as located at the center of the X-axis and at Y=20.

2. An instruction is associated with the label to center it horizontally on the reference point and locate it a distance 2.75 character heights below the reference point.

The preceding description of how to locate one label may seem complex. Why not just move the cursor to the starting location, perhaps (10,10), and draw the label? There are three problems with dictating the location of the label as the graph is created. First, the code does not know how long the label is. One label may be "Time" and another label may be "Friction Coefficient". An absolute location that centers one label will not center a label of different length. It is possible, however, to write the code to length-independent; it moves to the center of the axis, calculates the length of the label, moves a distance away from the center that corresponds to half the label's length, and then draws the label.

The second problem is that the code is device independent, but character size depends on what display is used. A low resolution monitor displays characters that are much bigger than those on a high resolution monitor. Therefore, a long label may span 50% of one monitor and only 10% of another. This has two implications. First, deciding on the location of a label must account for the type of monitor as well as the label's length. Second, two labels may overlap and be unreadable on one monitor but not overlap on another. The former may be solved by including the size of the character on the current display in the calculation of its location. The latter difficulty can be solved by looking to see if the label is going to overwrite anything, and not drawing it if it will.

The third problem is that the size of the graph is unknown. If there is one graph on the screen, it will cover the screen. If there are four graphs on the screen, then they will be much smaller. The size of the graph affects label placement just as the resolution of the screen does. Now calculating the label's location must include not only its length and the screen resolution, but also the size of the graph on the screen.

Plotting the points on the graph involves many of the same considerations as placing a label. The location of the curve on the screen depends on the screen's resolution and the number of plots on the screen.

As the complexities of creating device-independent code that displays multiple plots were revealed, it became obvious that a sophisticated approach to the graphics code was required. Writing straight code that accounted for every eventuality while remaining bug-free would be almost impossible (actually, prior experience on a different internal project at AMTI demonstrated the difficulty in creating good, straight-forward graphics code).

A design was developed that separates definition of the plot from its display on the screen. A plot can be totally defined but not displayed and then a separate routine displays the graph. The definition of the plot occurs without knowledge of the type of screen or how many plots will be shown on the screen. The graph is displayed by a routine that knows nothing about the graph but does know all the details about the screen, including its resolution and how many graphs will be shown.

The screen manager is responsible for placing objects (elements of a graph) on the screen. It is told how many graphs are to be displayed, and it is given a list of the graphs. It also has access to details about the display, such as the display's resolution, whether it is a CRT or printer, how many colors it can show, etc. When the screen manager shows the graphs, it uses the information about the display and the number of graphs to display each plot correctly.

The graphs are regarded as a list of objects such as lines and labels. Each object has information, such as its color, associated with it. Most of the information is specific to the type of object. For instance, the information for a line is its start and end points, and the information for a label includes its location, size, contents, orientation, and specifications for its placement. Each object also has two functions associated with it. One is the function that draws the object (for instance, one function draws a line, another draws a text string). The other function is one that destroys the object, freeing the memory for other uses. The functions are associated with the object using function pointers (ie. a pointer for the drawing-function is attached to the object so the screen manager can call that function to draw the object).

The structure definitions that define the objects and their relationships are held in the FW.H file in the appendix.

Each graph is actually a linked-list. A linked-list is a chain of objects in memory. Each link in the chain holds a pointer to the next link. The starting location of the list is known, and the list can be traversed by jumping from the first link to the next link, and then on to all subsequent links. These lists are "singly-linked", which means they can only be traversed from beginning to end (the links do not have pointers to the preceding links). Linked lists are used because they use memory efficiently. Only the memory needed to hold the list and its objects is reserved at any time.

A set of functions for linked-list management were created. They are held in the file LL.C and the structure definitions are held in LL.H. Each node in the list has two elements. One element is a pointer to the next node in the list and the second element is a pointer to the object associated with the current node.

A list is initialized by calling the function create_list(). This function reserves memory for the first three nodes (links) in the list. The first two nodes retain information about the list, the third node is the start of the useful nodes. The pointer to the "object" for the first node actually points to the current end of the list, allowing other functions to find the end of the list without traversing the entire chain. The purpose of the "object" for the second node is discussed later. The "object" of the third node is set to NULL, meaning it doesn't exist, and the pointer to the node following the third link is also set to NULL since there are no more nodes yet. The address of first node is returned as the address of the list.

The function add_link() adds a new link to the chain. It receives two parameters from the calling function. One parameter is the list to work with (passed as the address of the first node) and the second parameter is a pointer to the object to add to the list. This function does two things. First, it sets the "object pointer" for the last link to point to the new object. Secondly, it adds a new link to the chain by reserving memory for it and setting the former last link to point to the new last link and by setting the "object pointer" in the first link to point to the new last link.

The function traverse_list() travels through a given list and returns pointers to the objects in the list. The first time it is called for a given list, it returns a pointer to the object pointed to by the third node (the first active node) in the list. The next time it is called, it returns a pointer to the fourth node. When it reaches the end of the list, it returns NULL. It can be set to traverse the list again by calling the reset_list() function. This function uses the "object pointer" of the second node in the list to retain its current position. Every time the function is called it returns the object pointed to by the pointer in the "object pointer" element of the second node in the list. Then it updates the second node to point to the next active node in the list. This function is important because it is the only means outside routines have of accessing information stored in the list.

The last important functions are the destroy_objects() and destroy_list() functions. Destroy_objects() traverses a list and frees all the memory pointed to by the "object pointer" elements of the active nodes of the list (all nodes after the second). The destroy_list() function traverses a list and destroys all memory associated with the list. The destroy_objects() function must be called before destroy_list(), otherwise the objects will be retained but be inaccessible after the list is destroyed.

The graphics routines use the linked lists extensively. A graph is a linked-list (Figure 2.4.2). Each graph is a chain of objects that are to be drawn on the screen. An object may be a line, label, or poly-line (line with many segments). Additional objects, such as arcs, may be added easily. Each node of the linked-list holds a pointer to one object in the graph. For instance, one node may point to the x-axis label, the next to the y-axis label, the next to the x-axis line, and the next to a tic-mark label. The order of the nodes has no significance.

Graph-definition functions establish the linked-list that holds a graph. An outside function calls these functions and passes them information about the data, such as pointers to arrays of the values, labels for the axes, a title for the graph, and instructions (one instruction might be to draw the graph according to the right-hand-rule). The graph-definition functions then take the information and convert it to a series of objects that represent the graph. Eventually, the list of objects will be sent to

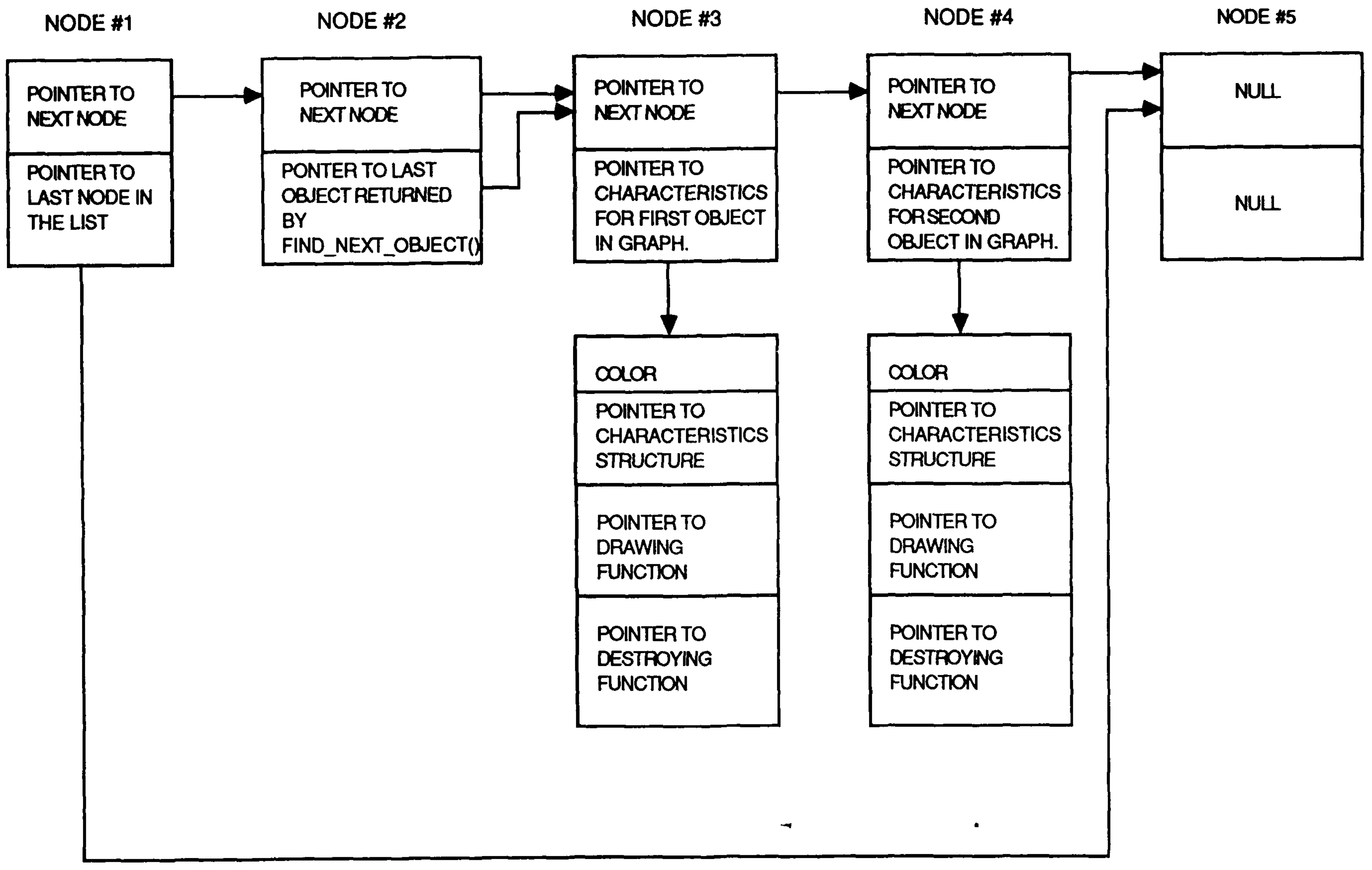

NODE #1
POINTER TO NEXT NODE
POINTER TO LAST NODE IN THE LIST
NODE #2
POINTER TO NEXT NODE
PONTER TO LAST OBJECT RETURNED BY FIND_NEXT_OBJECT()
NODE #3
POINTER TO NEXT NODE
POINTER TO CHARACTERISTICS FOR FIRST OBJECT IN GRAPH.
COLOR
POINTER TO CHARACTERISTICS STRUCTURE
POINTER TO DRAWING FUNCTION
POINTER TO DESTROYING FUNCTION
NODE #4
POINTER TO NEXT NODE
POINTER TO CHARACTERISTICS FOR SECOND OBJECT IN GRAPH.
COLOR
POINTER TO CHARACTERISTICS STRUCTURE
POINTER TO DRAWING FUNCTION
POINTER TO DESTROYING FUNCTION
NODE #5
NULL
NULL

FIGURE 2.4.2 - COMPLETE LINKED LIST WITH TWO GRAPH OBJECTS

the screen manager for display.

There may be different types of graph-definition functions. Some functions may establish an XY plot, others a bar chart, and still others may create a three-dimensional view of the data. The functions differ in the types of graphs that they define, but they share the same capability of taking data and converting it to a series of objects for the screen manager to place on the screen.

The only definition module created for this project produces an XY plot. These functions are held in the XY.C file (a sample program at the beginning of the file shows how to use the functions).

Creation of an XY plot takes two steps. The first is to initialize the plot, and the second is to add data to the plot. Plot initialization occurs through the initialize_xy_plot() function. This function sets the bounds for the plot, defines whether it follows the right-hand-rule, and defines the labels for the axes and the titles. This function returns a pointer which is used for a plot ID. More than one XY plot may be active at a time (perhaps Temperature vs. Time and Force vs. Time are both active) so there must be a means to identify them. The plot ID serves that purpose.

The plot-initialization function starts the linked-list that holds the objects for the graph. It first intializes the list, then it adds objects to the list that are the framework for the graph. For instance, it defines and adds two lines to the list that are the two axes. It calculates the locations and lengths of the tic-marks and adds those lines to the list. It computes the labels for the tic-marks and their locations and adds those strings to the list. Then it computes the locations for the axis labels and the titles and adds those strings to the list.

When the location of lines and strings are computed, it is done in relative terms and instructions may be attatched to the object that tell the screen manager how to place it on the screen. For instance, the locations of all lines are defined in normalized coordinates (between 0 and 100) so the screen manager can map the lines to the graphs region on the screen.

Defining the location of strings is more complex. As discussed previously, the placement of a string on the screen depends on its length, the screen's resolution, and the size of the graph on the screen. Therefore, the graph-definition module cannot dictate the absolute location of a string on a graph; instead, it supplies a reference point for the string and provides a series of instructions about placing the string relative to the reference point. For instance, the label for the x-axis should be centered under the axis at a distance that leaves space for the tic labels. The graph definition module defines the location for the axis by providing the center of the axis as a reference point and giving instructions to center the label horizontally at a distance under the reference point that corresponds to 2.75 character heights. When the screen manager draws the string on the screen, it uses the instructions to calculate the actual location for the string given the specifics of the monitor and the size of the graph. There are other instructions available, such as center the string vertically, right-justify it, left-justify it, and leave space horizontally between the string and the reference mark. An instruction is also provided that doesn't allow the string to be written if it will overwrite something on the screen. This instruction is used for tic mark labels because they often overwrite each other when the monitor's resolution is poor or the graph is small.

The add_polyline_to_plot() function adds data to the graph. A polyline is a series of line segments that define a curve; these segments are defined with an array of the X coordinates of the points on the curve and an array of the Y coordinates. The graph ID, pointers to the arrays, the number of points in the arrays, and the line color are passed to the add_polyline_to_plot() function. This function takes the information and converts it to a polyline object that is added to the graph's linked-list. More than one polyline can be added to a graph, and if the end of one polyline corresponds to th start of another they appear as one continuous line on the screen. This allows data to be displayed in pieces by adding a line to the graph, displaying it, and then adding another line to the chart and displaying that one.

Once a graph has been initialized, it can be shown on the screen by invoking the screen manager. The function add_graph_to_screen() adds a graph to the list of graphs that are to be shown on the screen. Then the graphics() function is called to intialize the monitor to graphics mode and set parameters for the screen manager. Now, all the active graphs may be drawn by calling the draw_screen() function. This function places all the graphs on the screen so they may be seen by the user.

When the screen manager draws a graph, it traverses the linked-list and processes each object in turn. When the entire list has been traversed and the objects drawn, the graph is complete on the screen.

The screen manager draws all graphs on the screen by first defining the location of a graph and then drawing it. The locations of the graphs are predefined and depend on how many graphs there are. If one graph is to be placed on the screen, it covers the entire screen. If two graphs are to be shown, they are placed in the upper and lower halves of the screen. For three or four graphs, the screen is broken into quadrants and one graph is placed in each quadrant (the lower-right quadrant is left blank when three graphs are drawn). For each of the cases, the coordinates of the corners of the graph windows are defined explicitly. These corners are then mapped to the coordinates of the normalized screen so that the objects, which are also defined in normalized-screen coordinates, will be drawn correctly.

The function draw_screen() in SCREEN.C displays all the graphs. For each graph, it first defines the locations of the upper-left and lower-right corners, then it draws the graph. The screen manager draws the graph by traversing its linked-list with the find_next_object() function. That function returns a pointer to one of the objects in the graph (which has the form defined by the object_s structure template in the FW.H file). As discussed previously, the elements of the object are its color, characteristics, drawing function, and destroying function. The screen manager calls the drawing function and passes it the object's color and characteristics. The drawing function uses the characteristics to place the object on the screen.

For instance, if the object is a line the elements of the line's object are its color, a pointer to its characteristics (a structure with the line_s template), a pointer to the draw_line() function, and a pointer to the destroy_line() function. The screen manager draws the line by calling the draw_line function via its pointer, and passing it the color and the pointer to the line_s structure. The form of the call is

```
(draw_func)(color,characteristics).
```

The function is defined by:

```
int (*draw_func)(unsigned color,void *characteristics)   (see graphs.h).
```

The declaration and calling of the drawing-function contains no information about what the object is. That information is held in the function that is actually called through the draw_func pointer.

In the case of the line, draw_func points to the draw_line function (see graphs.c). When this function gets control, it converts the characteristics pointer from type (void *) to type (line_s *). Then it uses the information in the structure to draw the actual line and returns to the screen manager.

The screen manager then uses find_next_object() to get a pointer to the next object in the graph and processes it just as it processed the line. If the next object is a label, then the draw_func pointer points to the draw_string() function. This function converts the characteristics pointer from type (void *) to type (text_s *) and uses that information to draw the string.

The screen manager proceeds in this manner through the entire graph. When the graph is complete, it moves on to the next graph if there is one. Otherwise, it returns to the calling program.

When a graph is no longer needed, it must be removed from memory. Graph removal requires several steps. First, the list must be traversed (using find_next_object()) and the function pointed to by the "destroy function" pointer. This function frees any secondary memory associated with an object. Some objects, such as lines, may not use any secondary memory so the function won't do anything. Other objects, such as polylines, do use secondary memory and the only way to free it is by calling the "destroy function" pointer. Then the list must be traversed again and the memory associated with the object's characteristics and with the object structure for the node must be freed. Finally, the memory used by the list itself must be freed. The function end_graph() (in GRAPHS.C) frees all memory associated with a graph using the preceding process.

2.4.5 Commercial Data-Acquisition Code

Although the software described in the preceding sections was designed, coded, and tested successfully a substantial amount of additional code was required to complete a full-scale customized data-acquisition, control and analysis package for the tribometer. The resources required to complete the project would exceed those available, so an alternative approach to the software was sought.

Rather than create a totally customized package for the tribometer, it was decided to use a commercial data-acquisition package and adapt it to the tribometer. Although the final package would not be as simple to use as a totally custom system, the development time would be much less and the final package would actually be more flexible than custom software.

As discussed in a previous section, Labtech Notebook was selected and adapted for use with the tribometer. The package is a general-purpose data-acquisition ·package with many provisions for automating and customizing the system.

Labtech Notebook's strength is data-collection. It has all the capabilities required for collecting data from the tribometer, and the capabilities are well documented in its user's manual. It does not do well, however, at immediate control of the output signals. There are no provisions for allowing the operator to turn digital signals on and off or easily set the level of an analog output signal, however Notebook does have provisions for executing outside programs while it operates. These programs can interact with Notebook if neccessary to access the data-acquisition boards.

The control program, FW.EXE, was adapted to operate with Labtech Notebook. FW.EXE does most of its control using the Metrabyte DDA06 board, but it also reads analog signals and sets the two analog outputs on the Data-Translation DT2801A board. Notebook does not interact with the DDA06 at all, therefore FW.EXE can set the output signals and analog outputs for that board directly and they will remain set while Notebook acquires data.

Notebook does use the DT2801A, though, so FW.EXE cannot access it directly. If it does, Notebook resets the board and the data becomes invalid. There is a means for outside programs to use boards that Notebook also uses. This is the Real-time Access (RTA) package that may be acquired with Labtech Notebook. This package allows programs to interact with data-acquisition boards through a device driver. Access to the boards, therefore, is much the same as access to a data file. A program opens the real-time-access system as it would open a file. Then it reads data from the board and send commands to the board as if it were a file. When the procedure is finished, the stream is closed.

FW.EXE uses the RTA system to work with the DT2801A. It acquires analog data by sending a command to Labtech Notebook, through the RTA, to read channels 1-16 (these are Notebook channels, they are defined to correspond to channels 0-15 on the DT2801A). Notebook then reads data from RTA until it encounters an end-of-file mark, signalling all the data has been sent. The values obtained through the RTA are data for channels 1 to 16 that have been converted to the proper units.

Similarly, FW.EXE sets the levels of the two D/A converters by sending a command through the RTA to set the level of these channels. D/A channels 0 and 1 on the DT2801A board have been defined as channels 17 and 18 in Notebook. Therefore, FW.EXE sets the level of D/A channel 0 by sending the appropriate value for channel 17 in Notebook and it sets the level of channel 1 by sending the appropriate value for channel 18.

Notebook may easily be customized and has provisions for starting up secondary programs while it is running. These capabilities are used to make the FW program part of the Notebook's menu system. Labtech Notebook has a built-in language called Magic-L which allows the user to modify Notebook's actions. When Notebook starts up, it looks for a file called AUTOLTN.MG and executes any code in that file. Furthermore, when the GO.EXE program is started (this is the program that actually acquires data within notebook) it looks for a batch file called AUTONB.BAT and runs that batch file before it begins taking data. These two provisions allow the FW.EXE program to start-up automatically when the user presses "P" (for Program).

The menu system in Notebook may be modified by using the Magic-L language. The Magic-L statement

```
m_item( 8 ) : r_pointer := base user1
```

forces Notebook to execute the Magic-L statements in the file USER1.MG when the "P" key is pressed. This statement is placed in the AUTOLTN.MG file so it becomes active when Notebook starts up. USER1.MG holds the statements

```
COPY AUTONB.B AUTONB.BAT
DOGO
DEL AUTONB.BAT
```

The first statement copies the file autonb.b to autonb.bat. The second statement executes the GO.EXE program, and the third statement deletes autonb.bat. This three step process is required because GO.EXE executes the AUTONB.BAT file whenever it starts. Therefore, the batch file is started when the user presses the "G" (Go) key as well as when GO is executed through the "P" key. Since the user presses "G" to take data during a test, not to run FW.EXE, the batch file that starts FW.EXE should be present only when the "P" key is pressed. The three step process above first copies the AUTONB.B file to AUTONB.BAT, then runs GO.EXE (which executes AUTONB.BAT), then deletes AUTONB.BAT after GO.EXE is finished. This sequence only occurs when the "P" key is pressed. When the "G" key is pressed, GO.EXE runs normally.

The AUTONB.B file holds one command, which is: FW

This command starts the FW program which then takes over the screen while GO.EXE runs in the background. As discussed previously, FW.EXE interacts with GO.EXE to read signals from the DT2801A.

3.0 WEAR MONITORING

Pin-on-disc testing is widely used to investigate the tribological properties of materials. The two main parameters of interest are the friction and the wear of the test samples. The frictional properties are easily measured in real time with appropriate force instrumentation. Wear measurement, on the other hand, is generally performed after the test is complete by mass loss, profilometry, or optical measurements. The most sensitive and accurate results are obtained from profilometry measurements of the disc and optical measurements of the pin wear flat. The initial conditions required are a flat disc and a pin with a spherical tip. Obviously, only a measurement of the cumulative wear is obtained after the test is complete. A steady state measurement of the wear rate during testing would provide additional useful information. Real time wear measurements are difficult to perform, but a technique has been devised which could allow real time measurements of moderate amounts of wear during testing. Essentially, it involves measuring the downward travel of the pin as it wears. The following equations can be used to determine the worn pin volume:

$V = 1/3\,\pi h^2\,(3r-h)$ = volume loss

and $h = r\,[\,1-\cos\,(\sin^{-1}\,(d/2r))]$, depth of pin wear or vertical measurement

r = spherical radius on unworn pin

d = wear flat diameter

The quantity h is measurable during a test, and the quantity d is measured after a test. The resolution obtained by measuring h is considerably lower than that obtained by measuring d. In addition, it is assumed that all of the wear occurs on the pin. This is generally the case with similar materials because the pin wear is concentrated at one point and disc wear is distributed over the wear track. The wear can still be measured if the disc wears instead of the pin, but the above wear volume equation would be incorrect. The correct equation would be:

$V = \pi DA$, wear volume on disc

where

D = wear track diameter

A = $\pi r^2 \cos^{-1}$ ((r-h)/r)/360 - .5(r-h)(r sin (.5 $\cos^{-1}$ (r-h)/r), sectional wear track area

and in this case

h = wear track depth into disc

In cases involving comparable depths of wear for the pin and the disc, some combination of the two wear volume equations must be used.

The accuracy of the on-line wear measurement system will be examined next. The measurement of h is the only unknown quantity which is needed to determine wear (assuming knowledge of whether the pin or the disc wears).

The entire tribometer structure including the specimen holders is designed to be rigid and undergo only elastic deflections under load. The vertical stiffness of the transducer module is approximately 60,000 pounds per inch (~10^7 N/M). A one pound (4.48N) normal load will therefore deflect the transducer 17 microinches (.42 microns). The remainder of the specimen support structure has a roughly equivalent stiffness which brings the total compliance down to 30,000 lbs/in (~5 x 10^6 N/M). Only the support structure deflections appear as axial wear to the displacement measuring systems due to the LVDT mounting configuration. Generally, most tribometer testing is performed at a constant load. Deflections due to normal loads should produce only small (<1 micron) errors in the wear monitoring system.

Thermal expansion accounts for most of the possible errors in the displacement monitoring system. The displacement transducers cannot differentiate between wear and thermally induced changes in length. Isothermal operation is therefore required for the highest accuracy. Thermal equilibrium or steady state conditions should be reached before starting an elevated temperature test.

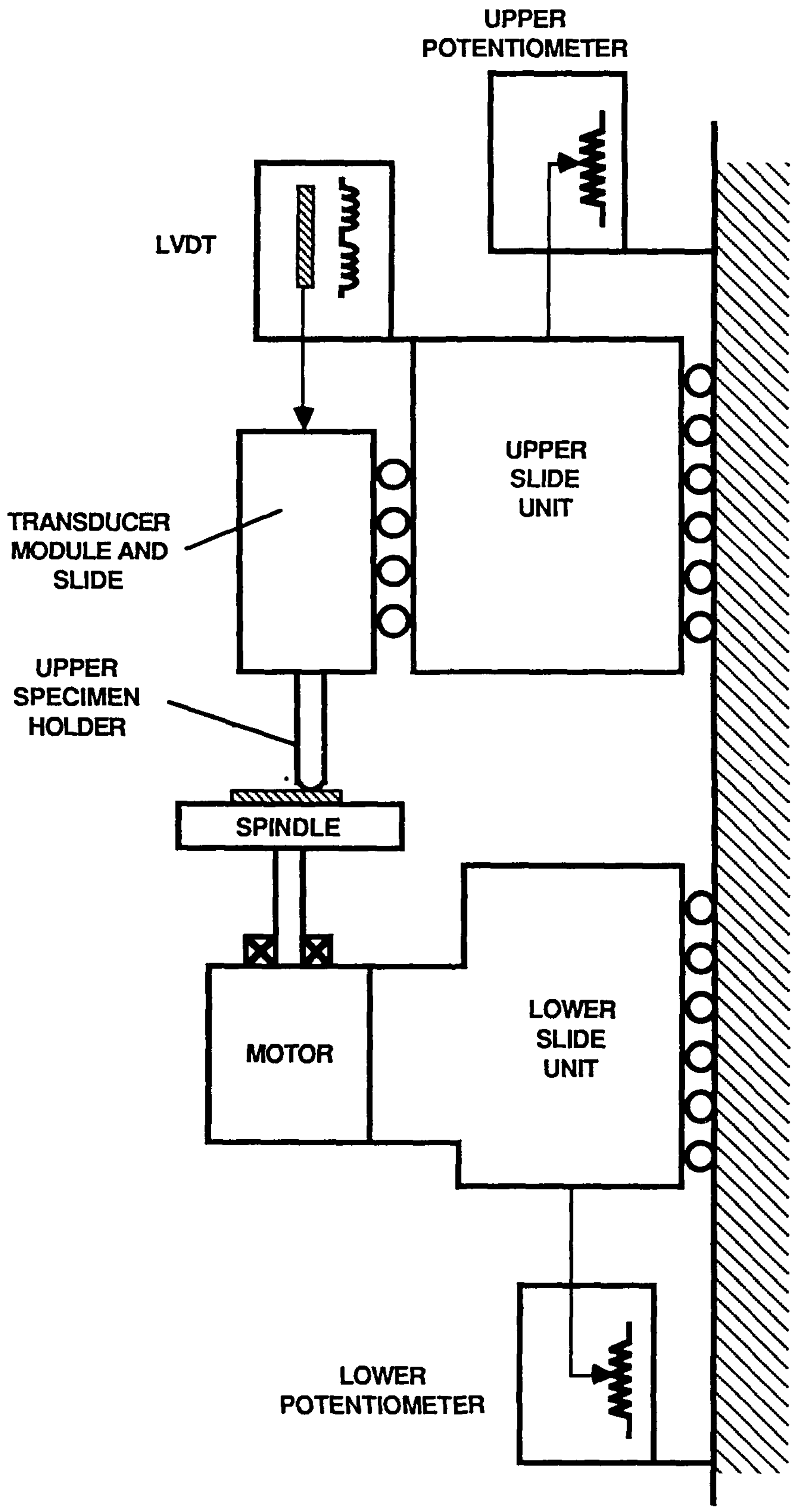
The entire vertical mounting plate of the tribometer is a rigid two-inch-thick aluminum plate. Water cooling channels machined into the plate keep it cool for elevated temperature testing. The water flow rate is approximately 0.7 gpm. At 1000 watts of cooling the outlet water temperature rises 10^{o}F (5.5^{o}C). The aluminum plate is reasonably isothermal, so it equilibrates at the median water temperature. The change in length of twenty inches (50.8 cm) of the aluminum baseplate is .00027"/^{o}F (12 microns/^{o}C).

Good control over the cooling water temperature is therefore required. The standard closed loop refrigeration system is a remote chiller made for drinking water or photographic use. It is acceptable for cooling purposes but tighter temperature control would reduce errors due to baseplate expansion. The actual water temperature control specifications are not published, but are estimated to be 5^{o}F (3^{o}C). The thermal mass of the tribometer (>300 pounds or 136 Kg of aluminum) smoothes out temperature fluctuations but the baseplate temperature will still vary as a function of cooling load. The variation is smaller for greater cooling loads because the chiller cycles on and off faster.

Refrigerated recirculating heat exchangers for cooling instruments are available as standard items (NESLAB Instruments) with $\pm$.1^{o}C temperature stability. These units would provide better control over cooling water temperature than is currently obtained.

A block diagram of the tribometer is shown in Figure 3.1. There are three slides on the machine; two large motor-driven slides are used to move the specimens into and out of the furnace and the third smaller slide is mounted on the upper large slide to provide small motions for applying the normal load. The two large slides are locked during testing and linear-potentiometer transducers provide an indication of their position. The force transducer is mounted on the smaller slide and the specimen holder position is sensed by an LVDT. All three position sensors are used to determine pin wear.

The motor-driven slides have a large travel and therefore the linear potentiometers need a correspondingly large range. An output of .8 in/volt (2.03 cm/volt) is obtained from the linear potentiometers which allows a relatively coarse position determination to be made. Assuming the use of a $\pm$10 volt 12 bit data acquisition system, the voltage measurement resolution is 5 millivolts which equals .004 inches (.01 cm) of slide travel. This is two orders of magnitude coarser than required for wear monitoring. However, the large slides should not move during a test because they are automatically locked into position by means of lead screw shaft brakes. This prevents them from moving,



Block Diagram of Tribometer Slides
Figure 3.1

but not necessarily on a micron scale. This can be verified and small motions compensated for in the following manner.

The required position of the motor driven slides during a test can be predetermined and set such that the linear potentiometer output becomes some definite value, for example, 2.790 volts. A separate voltage reference or setpoint of 2.790 volts can then be used to compare the actual output voltage to the setpoint voltage. The two voltages can be differentially summed, multiplied by 100, and the resulting output becomes .008 in/volt (.02 cm/volt). This give a 1 micron resolution on the motor driven slides. The resolution of the plastic conductive film type linear potentiometers is theoretically infinite which allows this technique to be used.

The position of both motor driven slides can be determined relative to fixed setpoints with a resolution of 1 micron. Referencing the precision supply voltage for the linear potentiometers and the setpoint potentiometers from the same precision 10.000V $\pm$10 mv source will produce essentially no error for small supply voltage variations.

Temperature gradients on the linear potentiometers themselves will produce errors, but testing is required to determine their magnitude. This error applies to elevated temperature testing where furnace radiation may heat the potentiometers. Internal I^2R heating is negligible and uniform. Conditions of thermal equilibrium should minimize this error.

The small slide on which is mounted the force transducer has an LVDT which provides an output of .01 in/volt (.025 cm/volt). The measurement resolution is therefore approximately 1 micron. The associated LVDT amplifier has automatic zeroing which is convenient for setting an initial zero wear setpoint. The LVDT plunger is spring loaded and bears directly on the portion of force transducer which clamps the specimen holder. Deflections of the transducer are not seen by the LVDT.

By far, the portion of the specimen holders in the furnace produces most of the thermal expansion in the tribometer. Assuming an average heated length of 6" (15.2 cm), the thermal expansion amounts to 33 microinches/^{o}F (1.5 microns/^{o}C) at room temperature and close to twice that at 2000^{o}F (1093^{o}C). It can be seen that under truly steady state conditions expansion effects could be quite small, but the gross thermal expansion from room temperature to maximum operation temperature can be .1 inch or 2500 microns. This is enormous compared to the wear depth.

All specimen holders in the furnace are presently made of inconel 625. Lower expansion alloys are available but provide a much lower maximum operating temperature. The thermal expansion increases with temperature and below 100^{o}C the errors due to expansion can be reduced by probably a factor of ten.

Proportional control of the furnace is required to prevent temperature oscillations. The heating load in the tribometer is constant at a given temperature, but the rapid response and low energy requirements at low temperature (<500^{o}C) requires a wide proportioning band and probably a reduction in maximum furnace power. Halving the furnace supply voltage from 230 volts to 115 volts will cut the power by a factor of four for better low end control.

The temperature produced expansions are the major errors for elevated temperature testing. Room temperature testing, however, still has thermal expansion errors due to frictional heating of the test samples. Steady state frictional heating cannot be reached without sliding, so thermal equilibrium cannot be reached before starting a test. The amount of heat generated can range from a fraction of a watt to more than a hundred watts, depending upon the test. The low end is typical of most tests, and the heat generated is estimated to produce less than 10 microns of thermal expansion error. All thermal expansion errors appear as specimen growth or negative wear. The higher the test temperature the lower the effect of frictional heating on thermal expansion.

4.0 LUBRICANT DELIVERY SYSTEMS

Lubricants for this program can be divided into two main categories; liquids and solids. Gaseous forms of lubricants (e.g., water vapor, oxygen, etc.) form another category, and the tightly sealed and enclosed nature of the tribometer permits their evaluation rather easily. No additional effort is required to further facilitate the use of gaseous substances since control of the test atmosphere is already provided.

4.1 Liquid Lubricant Delivery

The main variable which characterizes liquids as far as a feed system is concerned is viscosity. (Material reactivity with parts of the feed system is also important but does not appear to present problems.) Liquids may have viscosities which range up to a level where they could be considered solids. In addition, they can exhibit Newtonian or non-Newtonian behavior; grease is non-Newtonian but can often be dispensed as a liquid. We will deal with liquids within a viscosity range which allows them to be pumped. This covers a very wide range, and materials which resist suction flow can be prepressurized before being fed into the pump inlet. This technique is used in hand grease guns with a spring loaded piston.

In cases where only very small quantities of lubricant are available it is important to locate the dispenser close to the specimens and utilize small diameter hypodermic tubing for the feed lines. The required injection pressures can therefore be considerable for high viscosity or non-Newtonian fluids.

Peristaltic pumps have been used for liquid feed systems on the tribometer in the past. Typically, the pump unit is located external to the tribometer and feed lines run several feet to the specimen area. Elastomeric tubing which is compatible with the fluid being pumped is required. This is one of the main drawbacks to pumping some of the more exotic lubricants which have not been thoroughly tested. The tubing can be easily replaced and therefore does not require cleaning. The pump rate is determined by the rotational speed of the pump and by the tubing size.

Only fair control over the fluid feed rate can be obtained because the output is pressure dependent in spite of the fact that persistaltic pumps are positive displacement type devices. Pressure is generally limited to about 20 psi. Peristaltic pumps are poor from an efficiency standpoint, which means a large torque is needed to drive even a small pump. It is difficult to fit a pump and drive in a small space. The main advantage provided by a peristaltic pump is the ease with which the tubing can be changed.

A special metering pump which is adjustable and capable of very low flows has been chosen as the liquid feed mechanism. It is essentially an adjustable stroke reciprocating piston pump. Valving is done by the piston itself which also rotates. The manufacturer's literature is provided in Appendix III. Each rotation of the pump shaft provides one stroke; 1/2 of the rotation is the suction portion of the cycle and 1/2 of the rotation is the discharge portion of the cycle. The wetted parts of the pump are alumina and Kynar (fluorcarbon PVDF) which make them resistant to most fluids. The pump is easily disassembled for cleaning. The small size of the pump and its low torque requirements allows placement of the pump adjacent to the furnace. A small pump output line can then be filled with only a few tenths of a cc of fluid.

4.2 Solid Lubricant Delivery

Feeding powdered solid substances into a system is considerably more difficult than liquid lubricant injection. The simplified solid analogy to fluid viscosity would be particle size. Unlike fluids, however, a given powdered substance can have a very wide range of particle sizes and the behavior of a powder depends strongly upon this distribution of sizes. The size effect can be stronger than material effects; powdered teflon would not necessarily feed easily even though it has low friction properties.

Most solid material handling equipment is designed to function with a particular type of material. One made to feed granular material would not necessarily feed fine powder. The angle of repose, which is the maximum angle from horizontal that a pile of material will attain, is an important factor in most applications. A chute at an angle greater than the angle of repose always allows the material to slide down its surface.

A powder feed system which precisely dispenses up to one cc of material was designed and built. It is shown in Figure 4.1. Essentially a one stroke piston plunger device, the piston is pushed up through the cylinder by a linear actuator. The stepper-motor-driven actuator chosen travels 0.002" (.005 cm) for each pulse. The total stroke is one inch (2.54 cm) and the bore is 0.281 inches (0.714 cm). Approximately

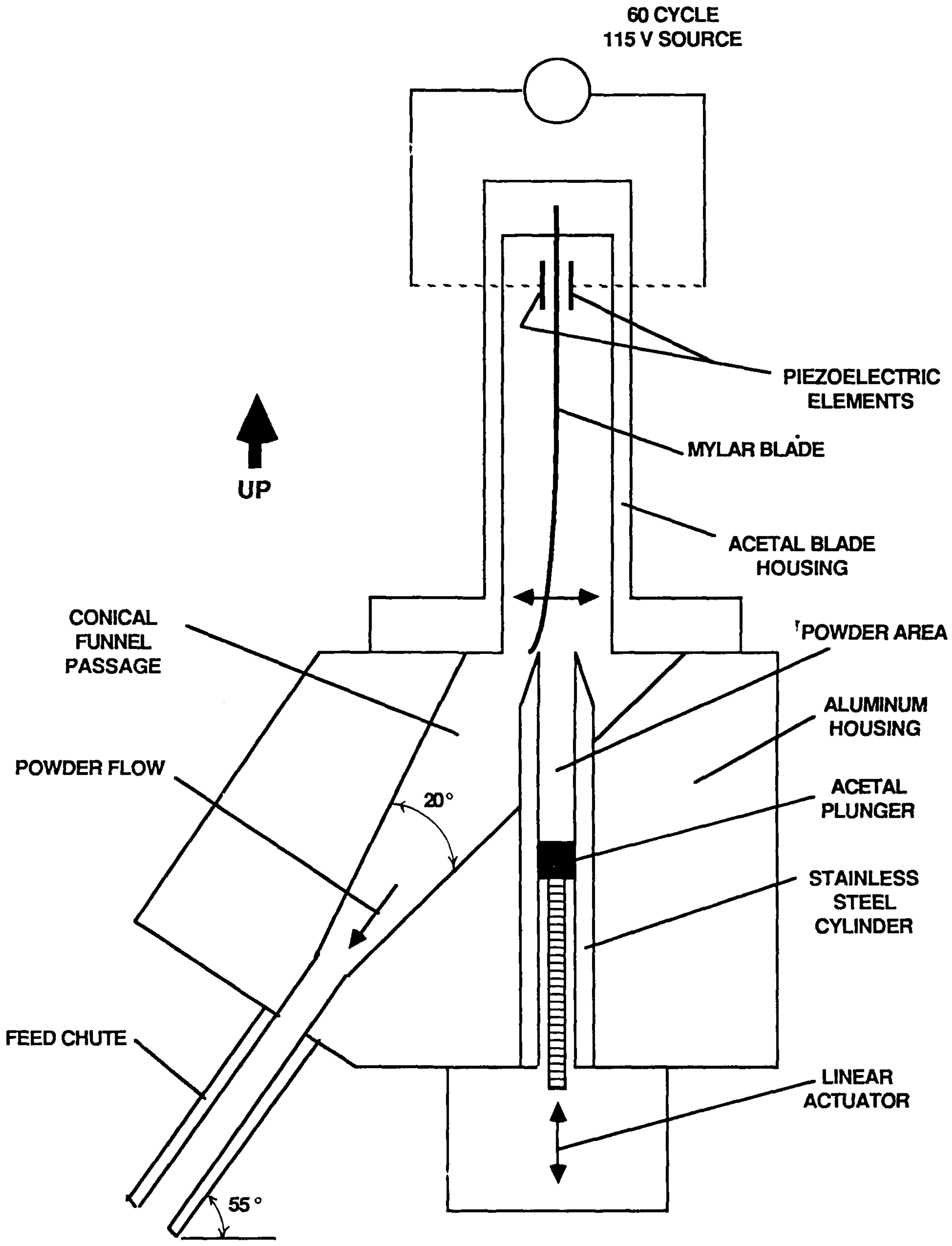


FIGURE 4.1 FEED SYSTEM

0.002 cm^3 of material is pushed out the end of the cylinder for each pulse of the actuator.

In order to provide a uniform feed rate which is independent of the angle of repose, a vibrating wiper was placed just above the top of the cylinder. This wiper is actually the main element of a piezoelectric fan. The wiper is a 3" long x 0.75" wide x 0.010" thick piezoelectric mylar strip which has two 0.75" square x 0.01" thick piezoelectric ceramic elements bonded to opposite faces at one end and is clamped further out at the same end. By supplying an oscillating excitation voltage to the elements at the mechanical resonant frequency of the mylar strip it can be made to vibrate with a significant end deflection. ($\pm$ 1 cm) The resonant frequency of the strip is approximately 60 hertz so it can be driven directly by AC line voltage. The impedance is high and @ 115 volts the power consumption is less than 100 mw. The force required to prevent blade motion is quite small, but sufficient to brush the powder particles from the top of the cylinder. Since no mounding of the powder can then occur, there is no change due to variations in the angle of repose of the powdered material.

Almost all angled surfaces which the powder contacts are 45^o or steeper. The exception is one very small saddleshaped area to the right of the cylinder. As long as the friction coefficient is less than one, the angle of repose is less than 45^o and all of the powder should fall down the feed tube.

Testing of the powder feed system was done with three materials: table salt, talcum powder, and Arizona road dust. The latter was chosen because it's particle size distribution is well characterized. It is often used as a test abrasive and a feed system for it would be a useful device by itself.

Table salt flows quite well through the system with the piezoelectric wiper <u>turned off</u>. A reproducible amount of salt is dispensed with every pulse of the linear actuator. With the wiper on, the top surface of the salt is not level with the top of the cylinder but is located approximately 1/16" (1.5mm) below the top. This is not due to air flow from the vibrating blade (it <u>is</u> made to be used as a fan) because in this enclosed geometry it does not function well as a fan. It occurs because the blade gives a downward vertical velocity to the particles, causing them to impinge on other particles and knock them out of the cylinder. The result is that the system feeds in spurts feeding into an elevated temperature furnace which is a good ignition source. Not all powders are ignitable, but many are and powders which react chemically at elevated temperatures to form good lubricants are probably ignitable.

One method of eliminating the problems encountered with finely divided particles is to combine them into larger particles. This can be done with many materials without any additional compounds in the end product; for instance water soluble powders such as confectioner's sugar can be bonded together with water and then dried and granulated. Small quantities of binders can be used in cases where the powder will not bond to itself. Relatively inert or heat-removable binders are available.

A simpler alternative to making a powder coarser would be to put it into a liquid carrier and pump it. The liquid feed system described earlier is well suited for pumping slurries. The liquid should not be detrimental to the end use of the powdered ingredient. Liquid carriers are frequently used to dispense powders such as mold release, molybdenum disulphide, and teflon in aerosol sprays.

4.3 Furnace Modifications

The original furnace had porous fused silica end plates. They were not impervious to liquids, and mechanically they were not well suited as structural members. Gold plated water-cooled aluminum end plates have been substituted for the silica plates. They are readily machined, unbreakable and very clean. Since they are actively cooled they do not heat up and the furnace cools rapidly when the power is turned off. Passages can be easily provided for lubricant injection and Luer lock fittings for hypodermic tubing can be attached to the top plate and can be easily removed. An inner quartz liner separates the windings from the specimens.

The aluminum end plates also allow a higher maximum operating temperature to be reached due to changes in the furnace windings. These changes would have been difficult without metal end plates. The furnace has been operated at $1200^{o}C$. Films of contaminants reduce the reflectivity of the gold film and they should be wiped off occasionally, especially for high temperature testing.

The ignition of the test lubricant has been mentioned as a problem which inevitably could occur. It was first thought that this would require some extinguishing technique such as an inert gas purge. However, the upper specimen holder has a gap between it and the furnace cover plate which is smaller than the typical flame quenching gap of .2 cm. This means that an internal flame will not escape from the top of the furnace. The inside of the furnace is reasonably impervious to heat from a small fire. A slow gas purge of the chamber might still be useful to displace oxygen from inside the furnace and put out a fire, but it could be introduced through the same line that would be used to feed a test gas into the furnace. Manual control of this type of extinguishing system is envisioned.

5.0 CONCLUSIONS

5.1 Software

1. It is possible to acquire data and supply control signals to the tribometer using IBM-PC compatible computers and commercially-available data-acquisition and control boards.

2. Using the commercial package "Labtech Notebook" for data-acquisition is preferable to writing a totally custom package. Notebook has all the required capabilities for data-acquisition, but it does not allow immediate control of the tribometer. It can be customized to provide the required control capabilities.

3. Creating a custom data-acquisition and control package for the tribometer is feasible. It would have the advantage that it is specifically tailored for the friction and wear machine, but early versions of the software would not be as adaptable as the commercial package and development of the software would require a significant commitment of resources.

4. An interrupt-driven data-acquisition module was developed. This module worked well and allowed a large degree of flexibility for monitoring the data collection.

5. A device-independent graphics module was created that allowed definition of graphs without knowledge of where the graph would be displayed and it allowed display of graphs without knowing any details about the graph. This system met the goals of providing a high-degree of flexibility while displaying single or multiple graphs reliably.

5.2 Wear Monitoring

1. Most of the errors associated with displacement measurement wear monitoring result from thermal expansion. Whenever possible, thermal equilibrium should be attained before starting a test to minimize these errors.

2. In most cases, it should be possible to detect axial specimen wear to within 10 microns. The accuracy will increase for low and moderate temperature tests and

even at room temperature a good closed loop cooling system should be used.

5.3 Lubricant Injection

1. Most liquid lubricants should be accurately meterable in small quantities by a commercially available metering pump. A wide range of flow rates is available from a single pump. No testing was performed on the pump because it appears to be suitable based on the manufacturer's literature.

2. A solid lubricant feed system was designed and a prototype was built. The unit functioned well with granulated material or materials which did not adhere to the walls. The wiper did not produce uniform flows and was found to be unnecessary. Materials with an angle of repose less than 45^{o} could be used in the feed system.

3. Some means for reducing the angle of repose or reducing the attraction of the powders to the walls of the feeder is necessary for the device to function with many fine powders. Converting the powder into a granular form or dispensing it as a liquid in a carrier fluid are two possible techniques.

4. The furnace design has been changed to permit lubricant testing by replacing the porous silica end plates with gold plated aluminum. This change also provided an easy way to connect lubrication fittings to the top furnace plate. Dangers associated with ignition of the test lubricants were judged to be minimal.

6.0 RECOMMENDATIONS

6.1 Software

1. Initial tests with relay boards and voltage sources show that the developed software works properly. However, the only way to discover subtle problems is by actual usage. Therefore, the software must be tested on an actual tribometer.

2. The software must be used by people who are not familiar with the software or computers. This experience will show how the package should be modified to make it easier to use and more reliable. The best testing method is to first use it extensively in-house and then at field sites on projects.

6.2 Wear Monitoring

Testing of the entire displacement monitoring system should be conducted on an actual tribometer. The sensitivity of the output to variations in water temperature, test temperature, and frictional heating needs to be investigated.

6.3 Lubricant Feed

The liquid lubricant pump chosen in this program should be installed next to the furnace of the tribometer. A slow drive motor controlled to make only complete revolutions should be used. The pump should be tested with a variety of fluids feeding into small diameter tubing. In cases where suction flow is insufficient to draw the lubricant from a reservoir a spring pressurized inlet feeder should be used.

A version of the powder feed system without the piezoelectric wiper but with a top cover should be installed on a tribometer. The outlet feed tube should be attached to the top furnace plate and an internal tube should direct the lubricant to the top of the disc specimen.

APPENDIX I

HIGH TEMPERATURE PIN-ON-DISC TRIBOLOGY APPARATUS

GENERAL

This apparatus has been designed and built to conveniently obtain friction and wear data on material test specimens up to 1200^{o}C. It consists of a rotating/oscillating spindle upon which the plate specimen is mounted and a special pin holder which provides the pin/disc contact load while measuring the normal and frictional forces. The pin/disc portion of the apparatus is enclosed in a special reflective film insulated furnace. The entire unit is enclosed in a sealed bell jar for atmosphere control. Movement of the specimens into and out of the furnace and control of the machine functions can be performed externally.

ENCLOSURE

The machine is provided with a glass/metal bell jar enclosure which allows the user to control the test atmosphere. A lead screw drive raises and lowers the bell jar over the machine; limit switches automatically stop the bell jar in the highest and lowest positions. A stainless steel cage serves as a thermal and protective barrier for the glass portion of the bell jar.

The bell jar allows the machine to be evacuated for purging and subsequent test atmosphere control. A mechanical roughing pump can be used to evacuate the system to 10 microns. A greater vacuum may be obtained with a turbomolecular or other type of high vacuum pump. Ports are provided for introduction of the desired test gas. Alternatively, the system can be operated under vacuum. The bell jar should not be raised while the system is at sub-atmospheric pressure. An electrical interlock disables the lead screw motor while under vacuum to prevent this from occurring inadvertently.

COOLING SYSTEM

A closed-loop chilled water system is used to cool and protect portions of the machine under conditions of elevated temperature testing. Distilled water, which has been specially treated with a biocide and corrosion inhibitor is circulated throughout the machine by a small pump. A thermostatically activated switch keeps the coolant at the desired temperature by controlling the refrigeration unit. The temperature can be set by a control on the chiller; 20^{o}C is the factory set point. An internal relay prevents heater power from being applied unless the chiller system is on. A pressure switch and an overtemperature switch also ensures that the machine is properly cooled during operation. A visible flow of water should always be observed in the flow meter when the coolant system is on. The chiller itself cycles on and off depending upon the heating load.

The cooling loop should remain on for a cooldown period even after heater power is turned off. Chiller power should not be removed while at elevated temperature or damage to the machine may occur.

FURNACE

A special gold reflective film-insulating furnace provides testing temperatures of up to 1200^{o}C. The furnace is wound with a Kanthal heating element and draws approximately 2700 watts at 230 volts. This type of furnace was chosen for its low thermal mass, rapid response, cleanliness and transparency at elevated temperatures. The top and bottom plates of the furnace are made of gold plated aluminum with water cooling and the inner tube is made of quartz. Contact with the glass shell of the furnace should be avoided while at high temperatures; the bell jar provides an excellent protective barrier in its lowered position.

Elevated temperature tests should be conducted with the upper and lower specimen holders in position before the heater power is applied. The specimen holders seal the openings in the furnace and prevent convection and radiation losses. In addition, thermal equilibrium should be reached before testing begins in order to minimize drift of the force transducers. The transducers should be zeroed prior to starting the test and periodically during a test.

The specimen holders thermally expand on the order of 1/16 inch along their axis at elevated temperatures. Re-adjustment or raising of the upper slide may therefore be necessary to prevent contact between the specimens before applying the test load. This is necessary only if the specimens were separated by only a small (.03") gap before heat up.

All elevated temperature tests should be conducted with the bell jar lowered to prevent accidental contact with hot surfaces and to provide a shield against possible breakage of any components.

SPINDLE CONTROL

A DC servo motor with tachometer and rotary potentiometer feedback controls the water-cooled spindle. Front panel controls allow either remote (computer) or front panel control of the motor. Rotary or oscillatory motion may be selected. In the rotary mode of operation the rotary potentiometer is not used for control although it still continuously senses the absolute shaft angular position. In the oscillatory mode of operation the rotary potentiometer is used to provide a feedback signal which the PWM (pulse width modulated) servo amplifier compares to a command signal. The feedback signal (shaft position) therefore follows the command signal. Note that the rotary potentiometer undergoes a step change from -10 volts to +10 volts in the vicinity of $\pm 180^{o}$ of shaft position. Oscillatory spindle control will not function properly if this discontinuity is crossed and the system may need to be reset by turning off either the power or the enable switch.

The function of each switch is described as follows:

POWER-ON/OFF	Turns on the main power to the PWM Servo Amplifier and therefore to the motor. In the event of overload the PWM Servo Amplifier will shut down without damage and can be reset by turning off its power for a few seconds.
REMOTE/PANEL	Provides either panel control or remote control of the spindle. In the remote position all other switches except for POWER are

	disabled and control is performed by computer inputs. All other switches are active when PANEL is selected.
ROTARY/ OSCILLATORY	Selects a rotary or oscillatory mode of spindle operation.
CW/CCW	Selects the direction of rotation in the ROTARY mode of operation.
SINE/TRIANGLE	Selects a wave-form shape which is either a sinusoid or a triangle wave in the oscillatory mode of operation.
ENABLE/OFF	Controls the output power stage of the PWM Servo Amplifier. This control should be used to start and stop the spindle motion.

The thumbwheel switches perform the following functions with the remote/panel switch in the PANEL position.

ROTARY SPEED	Allows speed selection from zero to approximately 1000 rpm in one rpm increments while in the ROTARY mode.
OSCILLATION FREQUENCY	Allows the oscillation frequency to be set between approximately .1 Hertz at a 000 setting to 5 Hertz at a 999 setting. Note that the allowable high frequency operation limit depends upon the amplitude and offset of the oscillary motion.
OSCILLATION AMPLITUDE	Allows the peak to peak oscillation amplitude to be set from zero (000) to $\pm178^{\circ}$ (999). Note that the highest amplitude can only be obtained with a zero degree offset (setting 500) and a slow oscillation frequency.
OSCILLATION OFFSET	Allows the DC average value of the waveform to be set between -178° (setting 000) and $+178^{\circ}$ (setting 999). A setting of 500 provides 0° offset and the largest amplitude and frequency capability.

An LED rpm indicator is provided which displays spindle speed. The DISPLAY panel lower LED selector switch can be set to display the rotary shaft position ($\pm$10.00 volts for $\pm178^{\circ}$ of rotation).

For external monitoring the tachometer provides $\pm$10 volts for $\pm$1000 rpm.

The entire spindle assembly is mounted on a linear cross roller slide actuated by a motor-driven lead screw. The spindle can be therefore easily moved into and out of the furnace. The slide motor is a synchronous stepper which can be stalled without damage but should not be routinely stalled to avoid reduced bearing life (see SLIDE CONTROLS).

PLATE - SPECIMEN HOLDER

The plate-specimen-holding portion of the spindle can be finely adjusted for axial runout by four differential-pitch brass screws located on the lower section of the rotating assembly. A dial indicator should be used for this purpose; as shipped from the factory the spindle has less than ±.0002" of axial runout. Radial runout is unimportant.

The plate specimen is held down by a clamping plate with four outer screws which attach to spring-loaded nuts in the baseplate. This allows for thermal expansion and keeps the plate specimen secure over the operating temperature range. A center screw is also provided for center mounting. Various clamping plates can be used for different specimen sizes.

NORMAL LOADING/TRANSDUCER UNIT

The pin specimen is mounted in a special tubular collet assembly which is held in a two-axis water-cooled force transducer. The friction or tangential force on the pin and the normal force or load are measured simultaneously. The transducer is a strain gage device and contains two independent four-arm 700-ohm bridges. Each channel is connected to its own amplifier which has been preset to provide 1 volt of output per pound of load. (A lower gain of .1 volt per Newton is also provided, see manufacturer's Amplifier Manual.) The transducer has a rated load capacity of 100 pounds in the normal force direction and 50 pounds in the friction force direction.

The upper specimen holder is a collet type device which grips the specimen over a wide temperature range. Replaceable and removable three piece collets are used for each diameter size of specimen; both pin and ball specimens can be used. A movable back-up pin inside the holder prevents the specimen from moving axially and allows different length pins to be used. Ball specimens must be balanced at both ends of the collet and are therefore best used in pairs with an intermediate spacer piece if necessary. Belleville washers provide spring loading of the collet jaws; the tightening nut should be snugged with a wrench for a few turns to firmly clamp the specimen. The entire tubular assembly fits into the force transducer and is secured by the front thumb screw.

The loading mechanism is an electromagnetic actuator which can provide from zero to 5 pounds of load. Preload weights and removal of the counter balance springs can also be used to give a total load capacity of greater than 20 pounds. This preload is applied by attaching one or two pairs of stainless steel weights to the transducer housing base. This base is the top of a linear slide controlled by the electromagnetic actuator. The slide is spring-suspended such that some of the electromagnetic actuator force is required to extend the springs. In all cases, however, the force transducer will read the true forces on the pin and plate specimens.

The entire transducer and actuator assembly is mounted on another motor-driven slide to permit movement into and out of the furnace. It is essentially identical to the spindle assembly slide with the exception that its base can be moved radially to provide wear track diameters from 0 to 2.75 inches. This adjustment can be made by loosening the four nuts on the slide posts, adjusting the slide in or out, and retightening the locking nuts.

SLIDE CONTROLS

The upper and lower linear slide tables which move the transducer and spindle unit into the furnace are electrically controlled. Each table has an AC synchronous stepping motor driving a ball screw in conjunction with a fail-safe shaft brake to prevent overhauling. A linear potentiometer is used to indicate slide position for automatic feedback.

Under manual control each slide table can be moved up or down via separate toggle switches; under automatic control the slide table will go to a predetermined position indicated by the computer or thumbwheel switch.

The function of each switch is as follows:

REMOTE/ PANEL	Allows the user to select either remote (computer) control or front panel switch operation. In the remote position the other associated switches are inoperative.
AUTO/ MANUAL	Selects automatic or manual control of the slide position. The automatic position is set with the associated thumbwheel switch.
LOWER	Lowers the slide for manual control.
RAISE	Raises the slide for manual control.
UPPER(LOWER) SLIDE POSITION	Setpoint for automatic control of the slide tables. The user should determine the appropriate setpoint from 000 to 999 for the required position. Each unit of the thumb wheel switch is approximately .0087 inches. Note that some settings of the switches are off the range of the slide tables. Upon reaching the setpoint the slide will automatically stop. A small amount of hysteresis is included to prevent hunting. In the event that some motor vibration occurs, the RAISE or LOWER switch should be momentarily jogged.

The display board lower LED selector switch can be set to display the actual slide position (LOWER SLIDE 1, UPPER SLIDE 1) or the amplified difference between the appropriate setpoint and the actual slide position. In this position (LOWER SLIDE 2, UPPER SLIDE 2) the output is approximately one volt per .009" of slide motion. Use of both the unamplified and amplified outputs of the slide position provides a high and low resolution output signal. This permits an accurate determination of specimen position to be made when used in conjunction with the optional wear measuring displacement transducer.

In order to perform a test the lower spindle unit should be raised into the furnace such that the specimen is in the middle of the furnace. The lower spindle gold-plated radiation shield should be at least partially in the furnace. The upper transducer slide unit should then be lowered into the furnace until the specimens are separated by a small (.06") gap. This is usually done by first making contact and then backing off on the upper slide. The electromagnetic actuator can move the upper specimen through the small gap and then apply the normal load. Care should be taken not to bottom out the small spring loaded slide and thereby apply a large force which may damage the two-axis transducer.

DISPLAY PANEL

Separate LED displays indicate the friction force, normal force, and friction coefficient during testing. The friction coefficient display can be switched to indicate friction coefficients from 0-1.000 or 0-2.000. Identical scaling of both force signals is necessary unless some other scaling of the friction coefficient is performed. Force signals should not be allowed to exceed 10 volts or an LED reading of 10.00. The internal divider which produces the friction coefficient divides the friction force signal by the normal force signal and therefore produces an undefined output due to division by zero when the normal force is zero. A valid friction coefficient signal is therefore produced only when the specimen is under load. The precision rectifier switch on the friction force amplifier produces the absolute value of the friction force signal; if the friction force is negative the precision rectifier should be switched on to provide a positive friction coefficient.

The lower LED selector switch connects one of nine signals to the LED display. The display reads up to $\pm$19.99 volts. The selections are listed as follows:

A	Connected to the optional upper specimen torque amplifier* output; user calibrated by applying a torque or differential moment along the pin specimen holder axis.
Wear Monitor	Connected to the optional wear measuring displacement transducer; user calibrated. Output depends upon amplifier settings; can be calibrated by displacing the transducer slide a known distance and determining the change in output.
Lower Slide 2	Displays the amplified difference between the lower slide setpoint and the actual slide position; .00875 inches/volt.
Lower Slide 1	Displays the lower slide position; .875 inches/volt output.
Spindle Angle	Displays the spindle angle; $\pm 178^{o}$ for $\pm$10.00 volts.
Upper Slide 1	Displays the upper slide position; .875 inches/volt output.
Upper Slide 2	Displays the amplified difference between the upper slide setpoint and the actual slide position; .00875 inches/volt.
Humidity	Displays the optional relative humidity sensor output; see manufacturer's instruction manual.
B	Displays the optional humidity sensor probe temperature; see manufacturer's instruction manual.

AMPLIFIER

The friction and normal force channels of the transducer each have their own amplifier. The optional upper specimen torque transducer and optional wear monitoring LVDT also have their own amplifiers. The amplifiers each have pushbutton taring and user adjustable gains, taring levels, filter settings, and balance adjustments.

Note that good zero balancing is most important for the friction force channel when the precision rectifier is on. This is due to the fact that any non-zeroed signal can be twice as large as indicated if the original non-rectified signal was negative.

Refer to the amplifier manuals for detailed operating instructions.

SERVICING

The apparatus requires very little periodic maintenance. The coolant can be changed every two years and replaced with treated distilled water. The lead screw drive is stainless steel and lubrication is unnecessary, but the linear bearing shaft is steel and should have a light film of oil to prevent rust. All motors are permanently lubricated. When available, the manufacturer's literature which has been provided should be consulted in regard to any questions concerning the separate components of the machine.

REQUIRED UTILITIES

Power: 230 VAC single-phase, 60 cycles, 25 amperes

APPENDIX II

SOURCE CODE

This is a list of the source code files created for the tribometer software package. A brief description of each file is here, and the source files are listed on a disk that is available from the authors.

The header files, which hold structure templates, definitions, and function declarations for the C code files are listed below.

FW.H Holds structure templates that are used throughout the FW code. The templates included are status_struct (holds status of digital entities such as the moter), the setpoints_struct (holds setpoint values), the channel_descrip struct (holds a description for each channel), and the dtoa_struct (holds conversion information for each D/A channel).

DAQ.H - Holds structure templates and definitions used in the realtime data-acquisition code. Includes templates for daq_control_s (holds information needed for data-acquisition but that does not have to be saved with the data set) and daq_descrip_s (holds information for data-acquisition that does have to be stored with the data).

LL.H- Holds the linked-list structure template link_s.

GRAPHS.H- Holds information for the graphics files. Includes templates for crt_desc (holds descriptive information for the CRT being used), graph_loc_s (tells where each graph goes on the screen), screen_list_s (holds a list of graphs for the screen), object_s (describes each object for a graph), text_s (describes an object that is a text string), line_s (describes an object that is a line), polyline_s (describes an object that is a polyline), and xy_plot_s (describes an XY plot).

The C code files are listed below with the functions that they hold. The first group of files contains the code for the FW control program.

FW.C

```
unsigned read_da_file(void)
unsigned read_ad_file(void)
unsigned get_delimited_string(FILE *fp,char *buff,unsigned length)
machine_control()
change_rotary_setpoints()
change_oscillatory_setpoints()
initialize()
```

FW_ST.C

```
change_value(char *descrip[],float *val_addrs[],float *min,float *max,char *units[])
```

FW_WIN.C

initialize_windows()
start_da_win()
update_da_win()
start_control_win()
update_control_win()
start_help()
stop_help()
void start_data_display(void)
void update_data_display(void)
void stop_data_display(void)

FW_ERR.C

disp_errorf(char *fmt,unsigned num, ...)

KBINT.C

void restore_kb_int()
void install_kb_int()
void (interrupt far kb_handler())
void get_control_key_status()

DDA06.C

set(unsigned device, unsigned action)
show_all_ports()
clear_all_ports()
set_dda06_outputs(unsigned *values)
initialize_dda06()

DT2801A.C

int initialize_dt2801a(int start_chan, int end_chan, int gain)
int wait(unsigned port, unsigned char n, unsigned char m)
set_dt2801a_outputs(unsigned *vals)

The next group of files holds the code for the realtime data-acquisition modules.

DAQ_INIT.C

unsigned initialize_daq(unsigned start_chan,unsigned end_chan, float period, unsigned preview,unsigned total_samples,unsigned trigger_method,unsigned trigger_level,unsigned trigger_channel, int (**call_funcs)(),unsigned auto_tare_chans,float *buffer,
float *conversions)

kernel select_kernel(float period)
void adjust_channels(unsigned kernel, unsigned start_chan,unsigned end_chan)
void terminate_daq(void)

DAQ.C

```
void show_dat(unsigned set)
```

DAQ_RT.C

```
unsigned take_zeros(unsigned channels)
unsigned go_daq(void)unsigned number_of_converted_sets,float *buffer)
void add_zeros(unsigned sc,unsigned ec,unsigned *dsp)
void convert_data(unsigned asc,unsigned aec,unsigned *dsp,float *conversions)
void store_data(unsigned sc,unsigned ec,float *buffer,unsigned set_number)
void call_user_funcs(int (**call_funcs)(unsigned), unsigned number_of_converted_sets)
int *get_data_set_ptr(void)
void adjust_trigger_level(void)
unsigned initialize_trigger_params(unsigned trigger_method, unsigned num_of_chans,
unsigned asc,unsigned trigger_chan, unsigned trigger_lev)
int initialize_buffer(unsigned sc, unsigned ec,unsigned preview, unsigned total_samples)
unsigned find_empty_seg(void)
void free_empty_seg(unsigned seg)
void set_timer(float t_set)
void reset_timer(void)
void install_timer_int(void (interrupt far *t_handler)())
void  restore_timer_int(void)
void initialize_dma_chip(unsigned page)
```

This group of files holds the code for the graphics modules.

LL.C

```
void reset_list(struct link_s *list)
void *find_next_object(struct link_s *list)
LL create_list(void)
void *add_link(struct link_s *list,void *object)
void destroy_list(struct link_s *list)
void destroy_objects(LL list)
```

XY.C

```
unsigned main(void)
unsigned y_axis_label(XY_PLOT xyp,char *str)
unsigned x_axis_label(XY_PLOT xyp,char *str)
unsigned title_graph(XY_PLOT xyp,char *str)
unsigned poly_line_abs(XY_PLOT xyp,float *x,float *y, unsigned num_pts,unsigned color)
unsigned add_polyline_to_plot(XY_PLOT xyp,float *xx,float *yy,unsigned
num_pts,unsigned color,unsigned type)
void mapatog(XY_PLOT xyp,float a,float b,float *x,float *y)
XY_PLOT initialize_xy_plot(float ax_min,float ay_min, float ax_max,float ay_max,int
rhr,char *x_label,char *y_label,char *title)
void end_xy_plot(XY_PLOT xyp)
unsigned draw_axes(XY_PLOT xyp)
unsigned draw_tics(XY_PLOT xyp)
void convert_to_label(float value,char *buff)
int num_tics(int span,int *tic_interval)
float adjust_span(float span)
```

GRAPHS.C

unsigned main(void)
GRAPH create_graph(void)
void end_graph(GRAPH graph)
unsigned add_line_to_graph(GRAPH graph,float x1,float y1,float x2,float y2, unsigned color,unsigned type)
void draw_line(unsigned color,void *characteristics)
void destroy_line(void *characteristics)
unsigned add_polyline_to_graph(GRAPH graph, float *xx, float *yy, unsigned num_pts, unsigned color, unsigned type)
void draw_polyline(unsigned color,void *characteristics)
void destroy_polyline(void *characteristics)
unsigned add_string_to_graph(GRAPH graph,char *str,float x, float y, unsigned action,float blanks,unsigned orientation,unsigned color)
void draw_string(unsigned color,void *characteristics)
void destroy_string(void *characteristics)
blank_region(float x1,float y1,float x2,float y2, unsigned color)

G_INIT.C

void time_out_handler(void)
unsigned graphics(int type,int enable_hp,char *g_dir,int action) hp_command(char *cmd,...)
change_to_graph_dir()
struct view_port *define_viewport(x1,y1,x2,y2)
free_viewport(vp)
set_viewport(x1,y1,x2,y2)
set_text_path(unsigned path)
inq_err()
set_color(i)
st_width(str)
st_height(str)
st_offset(str)
ln_abs(x1,y1)
mov_rel(x1,y1)
mov_abs(x1,y1)
set_world(x1,y1,x2,y2)
mov_tcurrel(x,y)
init_tcur(i,j,k)
set_textclr(i,j)
set_lnstyle(i)
ln_rel(x1,y1)
poly_lnabs(float *xx,float *yy,int num_of_points)
set_stang(ang)
set_crange(i)
set_drange(i,j)
ft_size(i,j)
ft_locate(i,j)
ft_color(i,j)
set_stclr(i,j)
set_stext(x,y,i)
set_text(i,j,k,l)

```
mov_tcurabs(x,y)
circle(float x) /* x is the radius */
b_text(char *str)
```

SCREEN.C

```
void clear_graphics_screen_list(void)
unsigned add_graph_to_screen(GRAPH graph)
void draw_screen(void)
void update_screen(void)
```

FW_WIN.C

initialize_windows()
start_da_win()
update_da_win()
start_control_win()
update_control_win()
start_help()
stop_help()
void start_data_display(void)
void update_data_display(void)
void stop_data_display(void)

FW_ERR.C

disp_errorf(char *fmt,unsigned num, ...)

KBINT.C

void restore_kb_int()
void install_kb_int()
void (interrupt far kb_handler())
void get_control_key_status()

DDA06.C

set(unsigned device, unsigned action)
show_all_ports()
clear_all_ports()
set_dda06_outputs(unsigned *values)
initialize_dda06()

DT2801A.C

int initialize_dt2801a(int start_chan, int end_chan, int gain)
int wait(unsigned port, unsigned char n, unsigned char m)
set_dt2801a_outputs(unsigned *vals)

The next group of files holds the code for the realtime data-acquisition modules.

DAQ_INIT.C

unsigned initialize_daq(unsigned start_chan,unsigned end_chan, float period, unsigned preview,unsigned total_samples,unsigned trigger_method,unsigned trigger_level,unsigned trigger_channel, int (**call_funcs)(),unsigned auto_tare_chans,float *buffer, float *conversions)

kernel select_kernel(float period)
void adjust_channels(unsigned kernel, unsigned start_chan,unsigned end_chan)
void terminate_daq(void)

DAQ.C

void show_dat(unsigned set)

DAQ_RT.C

unsigned take_zeros(unsigned channels)
unsigned go_daq(void)unsigned number_of_converted_sets,float *buffer)
void add_zeros(unsigned sc,unsigned ec,unsigned *dsp)
void convert_data(unsigned asc,unsigned aec,unsigned *dsp,float *conversions)
void store_data(unsigned sc,unsigned ec,float *buffer,unsigned set_number)
void call_user_funcs(int (**call_funcs)(unsigned), unsigned number_of_converted_sets)
int *get_data_set_ptr(void)
void adjust_trigger_level(void)
unsigned initialize_trigger_params(unsigned trigger_method, unsigned num_of_chans, unsigned asc,unsigned trigger_chan, unsigned trigger_lev)
int initialize_buffer(unsigned sc, unsigned ec,unsigned preview, unsigned total_samples)
unsigned find_empty_seg(void)
void free_empty_seg(unsigned seg)
void set_timer(float t_set)
void reset_timer(void)
void install_timer_int(void (interrupt far *t_handler)())
void restore_timer_int(void)
void initialize_dma_chip(unsigned page)

This group of files holds the code for the graphics modules.

LL.C

void reset_list(struct link_s *list)
void *find_next_object(struct link_s *list)
LL create_list(void)
void *add_link(struct link_s *list,void *object)
void destroy_list(struct link_s *list)
void destroy_objects(LL list)

XY.C

unsigned main(void)
unsigned y_axis_label(XY_PLOT xyp,char *str)
unsigned x_axis_label(XY_PLOT xyp,char *str)
unsigned title_graph(XY_PLOT xyp,char *str)
unsigned poly_line_abs(XY_PLOT xyp,float *x,float *y, unsigned num_pts,unsigned color)
unsigned add_polyline_to_plot(XY_PLOT xyp,float *xx,float *yy,unsigned num_pts,unsigned color,unsigned type)
void mapatog(XY_PLOT xyp,float a,float b,float *x,float *y)
XY_PLOT initialize_xy_plot(float ax_min,float ay_min, float ax_max,float ay_max,int rhr,char *x_label,char *y_label,char *title)
void end_xy_plot(XY_PLOT xyp)
unsigned draw_axes(XY_PLOT xyp)
unsigned draw_tics(XY_PLOT xyp)
void convert_to_label(float value,char *buff)
int num_tics(int span,int *tic_interval)
float adjust_span(float span)

GRAPHS.C

unsigned main(void)
GRAPH create_graph(void)
void end_graph(GRAPH graph)
unsigned add_line_to_graph(GRAPH graph,float x1,float y1,float x2,float y2, unsigned color,unsigned type)
void draw_line(unsigned color,void *characteristics)
void destroy_line(void *characteristics)
unsigned add_polyline_to_graph(GRAPH graph, float *xx, float *yy, unsigned num_pts, unsigned color, unsigned type)
void draw_polyline(unsigned color,void *characteristics)
void destroy_polyline(void *characteristics)
unsigned add_string_to_graph(GRAPH graph,char *str,float x, float y, unsigned action,float blanks,unsigned orientation,unsigned color)
void draw_string(unsigned color,void *characteristics)
void destroy_string(void *characteristics)
blank_region(float x1,float y1,float x2,float y2, unsigned color)

G_INIT.C

void time_out_handler(void)
unsigned graphics(int type,int enable_hp,char *g_dir,int action) hp_command(char *cmd,...)
change_to_graph_dir()
struct view_port *define_viewport(x1,y1,x2,y2)
free_viewport(vp)
set_viewport(x1,y1,x2,y2)
set_text_path(unsigned path)
inq_err()
set_color(i)
st_width(str)
st_height(str)
st_offset(str)
ln_abs(x1,y1)
mov_rel(x1,y1)
mov_abs(x1,y1)
set_world(x1,y1,x2,y2)
mov_tcurrel(x,y)
init_tcur(i,j,k)
set_textclr(i,j)
set_lnstyle(i)
ln_rel(x1,y1)
poly_lnabs(float *xx,float *yy,int num_of_points)
set_stang(ang)
set_crange(i)
set_drange(i,j)
ft_size(i,j)
ft_locate(i,j)
ft_color(i,j)
set_stclr(i,j)
set_stext(x,y,i)
set_text(i,j,k,l)

```
mov_tcurabs(x,y)
circle(float x) /* x is the radius */
b_text(char *str)
```

SCREEN.C

```
void clear_graphics_screen_list(void)
unsigned add_graph_to_screen(GRAPH graph)
void draw_screen(void)
void update_screen(void)
```

APPENDIX III

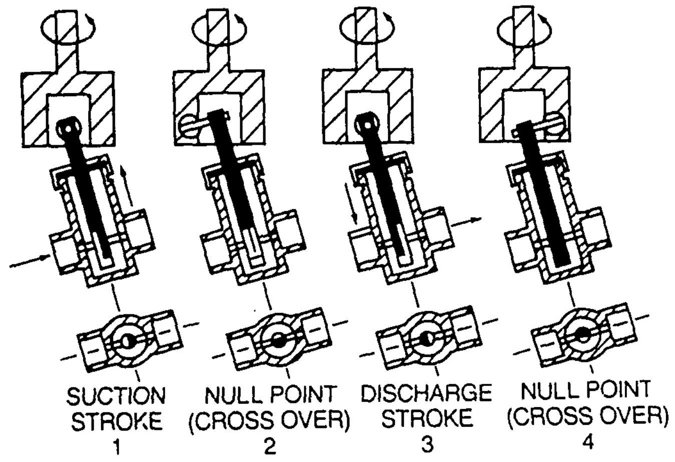
LIQUID LUBRICANT PUMP

The following pump has been selected to provide fine adjustable metering of liquid lubricants for the tribometer. It is rated to handle viscosities up to 8000 cp with a pressurized inlet.

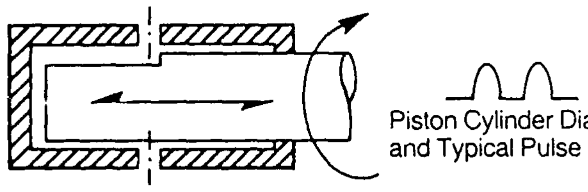
THE FMI METERING PUMP STORY

The FMI valveless pump lines are designed specifically for safe, accurate handling of liquids and gases in laboratory, pilot plant, production and OEM applications. The "Q" and "RH" lines are available with a variety of Pump Drive Modules and Pump Head Modules in various sizes and corrosion resistant wetted materials. The numerous combinations obtainable will meet the requirements of most applications.

OPERATION: The valveless pumping function is accomplished by the synchronous rotation and reciprocation of the piston in the precisely mated cylinder bore. One pressure and one suction stroke are completed per cycle. A duct (flat portion) on the piston connects the cylinder ports alternately with the pumping chamber, i.e., one port on the pressure portion of the pumping cycle and the other on the suction cycle. The mechanically precise, free of random closure variation valving is performed by the piston duct motion



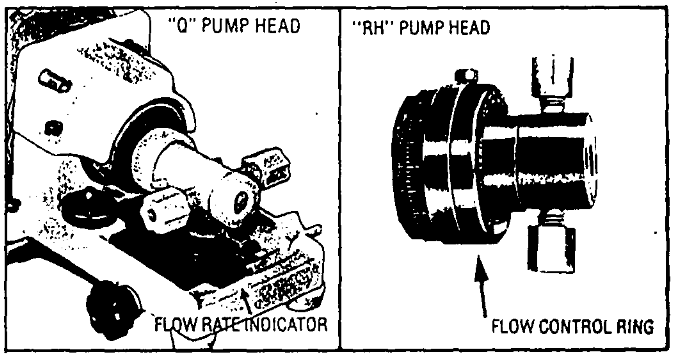
The Pump Head Module containing the piston and cylinder is mounted in a manner that permits it to be swiveled angularly with respect to the rotating drive member. The degree of angle controls stroke length and in turn flow rate. The direction of the angle controls flow direction. The reciprocation accuracy and positive valving of FMI pumps provide exceptional performance and dependability. For best pumping results select a pump head having a maximum flow rating as near the desired flow rate as possible.


Piston Cylinder Diagram and Typical Pulse Form

PRESSURE: In most FMI pump models, motor starting torque is the limiting factor in the stated pressure rating. Fluids such as oils, creams and gels that are good lubricants are more easily pumped than aqueous or "dry" fluids and therefore require less motor torque and may be pumped against pressures considerably greater than those given in the rating charts.

All pump head components are designed to withstand back-pressures up to 100 psig at room temperatures, though pump heads with kynar cylinder cases may exhibit some loss of pumping capacity at pressures over 60 psig.

FLOW RATE: FMI pump flow rates may be altered when operating or at rest. On the "Q" line pumps this is done by turning the flow control knob which moves the flow rate indicator along a fixed 20 unit scale linearly calibrated "10-0-10". The "10" meaning 100% flow rate in one direction, "0" meaning zero flow and the second "10" meaning 100% flow in the opposite direction. To improve the fine adjustment of the flow rates on the "Q" line there is an optional Dial Indicator Kit available. The "RH" line flow adjustment is accomplished by turning an easy-grip flow control ring graduated in 450 divisions from 0 to 100% flow.



ACCURACY: FMI pump accuracy is based on a simplified positive displacement mechanism. The valveless design provides reproducibility of better than 1% when handling medium viscosity fluids (50 to 500 centipoise). Aqueous solutions and light solvents work well but may exhibit some sensitivity (fluid slip) to variations in discharge head pressure. Gums, gels and non-abrasive semi-solids are handled with a high degree of accuracy . . . a direct result of the valveless design.

Viscous, tacky solutions, semi-solids and heavy slurries which tend to resist (cavitate) suction flow into a pump head can be handled with ease by selecting an FMI pump employing a relatively slow reciprocation rate.

The principal flow rate deviations of an FMI pump are fluid slip and stroke repetition rate. These two factors in turn are related to load factors such as viscosity, differential pressure, and drive motor voltage. When these factors are controlled, the FMI pump will handle most fluids with reproducibility of better than 0.1%.

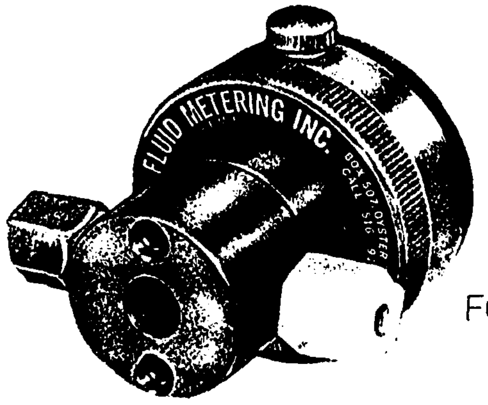
GAS PUMPING: Due to the valveless design of the FMI pump, "CKY" and "CSY" pump heads are able to perform acccurate gas transfers. With no valves to introduce random compression errors, gas sample flow in bagging, scrubbing and transit operation can be accurately preset on a basis of actual piston displacement.

IMPORTANCE OF CLEAN FLUIDS: While a certain amount of caution must be exercised in the use of abrasive fluids in any metering pump, the CKC and CSC pumps tend to be more tolerant of suspended solids than other metering pumps. To assure fluid compatibility, consult the Materials of Construction information on page 23 of this catalog.

FOR BEST PUMPING RESULTS: Select an FMI PUMP having a maximum flow rating as near to the desired flow rate as possible.

MOTORLESS PANEL MOUNT MODELS "RH"

RH



MODEL RH

"RH" FMI LAB PUMP JR. FOR TINY FLOWS

FOR OEM AND INSTRUMENT MANUFACTURERS

0 to 360 ml per minute
Up to 3600 rpm

From -10 to 100 psig
Only 2¼" panel space needed

The RH Lab Pump Jr. line is self-contained, low flow, valveless, positive displacement pumps that may be panel mounted and are extremely popular with any user needing a compact pumping unit such as instrument manufacturers. They have double row ball bearings and a power requirement of 10 inch ounces of torque from any rotational source such as pulleys, belts, chains, motors, or shaft coupling in response to rotating machine elements.

Flow adjustment is easily accomplished either while running or at rest, by turning the easy-grip flow control ring, graduated in 450 divisions. This provides accurate adjustment of 0.05 and 0.10 ml maximum per stroke with reversible flow direction accomplished by changing drive direction. Linear speed versus flow rate is from 0 to 3600 rpm with a stroke length adjustment of 0 to 100% for maximum flexibility of flow rate.

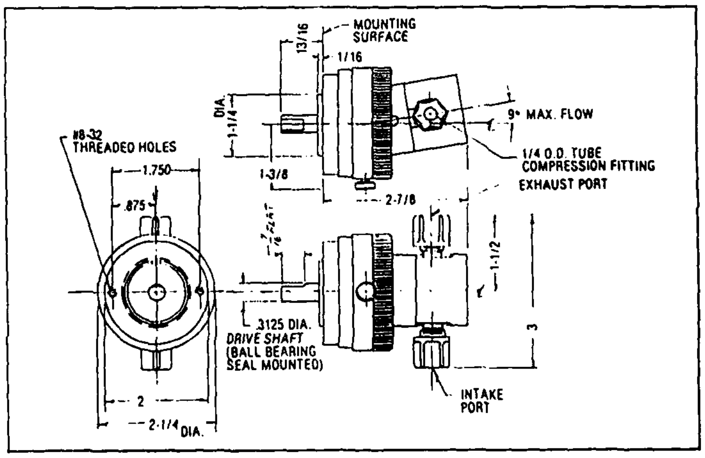
RH CAPACITY CHART THROUGH-PUT EXAMPLES OF RH PUMP HEADS MOUNTED ON STANDARD FMI PUMP DRIVE MODULES			
PUMP DRIVE MODULES	STROKES PER MINUTE	MAX. FLOW RATE RHOCKC (ml/min)	MAX. FLOW RATE RH1CKC (ml/min)
	1	0.05	0.10
RP-G6	6	0.30	0.60
RP-G20	20	1.00	2.00
RP-G50	50	2.50	5.00
QSYX, QSY	72	3.60	7.20
RP-G150	150	7.50	15.00
RP-G400	400	20.00	40.00
QD, QDX	1725	86.25	172.50
RHB, QB, RHV, QV	1800	90.00	180.00

The RH line has the added versatility of being able to be mounted on any of the Q line Drive Modules by use of the RH/Q Adapter shown on page 20 as well as being available with their own integral motor drives.

Dimensions: 2 1/4" O.D. x 3 1/2" long this includes a 3/4" long x 5/16" dia. drive shaft.
Shipping weight: 1 lb.

COMPLETE PUMP ASSEMBLY	MAX. FLOW RATES	PRICE
RHOCKC	0 to 0.05 ml/stroke	$360
RH1CKC	*0 to 0.10 ml/stroke*	
SELECTION	WETTED PARTS	CERAMIC KYNAR
GUIDE	MAX. TEMP	212° F

RH LAB PUMP JR. OUTLINE DRAWING



FLUID METERING, INC., 29 Orchard St., P.O. Box 179, Oyster Bay, NY 11771 (516) 922-6050 • Telex 5101001281 • Fax (516) 624-8261